Tampere University

ANTTI HIETANEN

Computer Vision for Robotics

Feature Matching, Pose Estimation and Safe Human-Robot Collaboration

Tampere University Dissertations 370

Tampere University Dissertations 370

ANTTI HIETANEN

Computer Vision for Robotics

Feature Matching, Pose Estimation and Safe Human-Robot Collaboration

ACADEMIC DISSERTATION
To be presented, with the permission of
the Faculty of Information Technology and Communication Sciences
of Tampere University,
for online public discussion,
on 15 January 2021, at 12 o'clock.

ACADEMIC DISSERTATION
Tampere University, Faculty of Information Technology and Communication Sciences
Finland

Responsible supervisor and Custos	Professor Joni-Kristian Kämäräinen Tampere University Finland	
Supervisor	Professor Minna Lanz Tampere University Finland	
Pre-examiners	Professor Patric Jensfelt KTH Royal Institute of Technology Sweden	Assistant Professor Juho Kannala Aalto University Finland
Opponents	Professor Patric Jensfelt KTH Royal Institute of Technology Sweden	Professor Juha Röning University of Oulu Finland

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

Cover design: Roihu Inc.

ISBN 978-952-03-1839-0 (print)
ISBN 978-952-03-1840-6 (pdf)
ISSN 2489-9860 (print)
ISSN 2490-0028 (pdf)
http://urn.fi/URN:ISBN:978-952-03-1840-6

PunaMusta Oy – Yliopistopaino
Joensuu 2021

PREFACE/ACKNOWLEDGEMENTS

The work related to this thesis was carried out at Tampere University (previously known as Tampere University of Technology), Finland, between 2016-2020. The thesis is a summary of the collection of my research papers which I published during my doctoral studies as a member of the Computer Vision Group and the Intelligent Production Systems Group.

First and foremost, I would like to thank my supervisor Joni-Kristian Kämäräinen for his guidance and support during my studies. He has proven to be a great team leader with endless source of crazy new ideas and true willingness to direct his students. I also owe my deepest gratitude to Minna Lanz who introduced me to the field of robotics and has been significantly involved in supervising my work. I would also like to give special thank Alessandro Foi for his effort and endless patience while guiding me in mathematical issues.

I want to thank all my colleges at the Computer Vision Group and the Intelligent Production Systems Group for creating a motivating work environment. We have had many fruitful discussions about research work but also important conversations of non-work related matters which have brought joy and laughter into my days at the university.

Last but not least I want to thank my family for supporting me unconditionally. I thank my girlfriend Pinja for standing by my side through the whole journey and her understanding when I had to stay long hours at the lab.

ABSTRACT

This thesis studies computer vision and its applications in robotics. In particular, the thesis contributions are divided into three main categories: 1) object class matching, 2) 6D pose estimation and 3) Human-Robot Collaboration (HRC). For decades, the 2D local image features have been applied to find robust matches between two images of the same scene or object. In the first part of the thesis, these settings are extended to class-level matching, where the primary target is to find correct matches between object instances from the same class (e.g. Harley-Davidson and scooter from the motorcycle class). The current benchmark is modified to the class matching setting and state-of-the-art detectors and descriptors are evaluated on multiple image datasets. As a main finding from the experiments, the performance of the 2D local features on class matching settings is poor and specialized approaches are needed.

In the second part, the local features are extended to 6D pose estimation where the 3D feature correspondences are used to fully localize the target object from the sensor input, i.e. to give its 3D position and 3D orientation. For finding reliable correspondences, two robustifying methods are proposed that exploit the input object surface geometry and remove unreliable surface regions. Based on the experiments, the relatively simple algorithms were able to improve the accuracy of several pose estimation methods. As a second study on the pose estimation category, the existing evaluation metrics for measuring the qualitative performance of an estimated pose are assessed. As a results, we proposed a novel evaluation metric which extends the current practices from geometrical verification to a statistical formulation of the task success probability given an estimated object pose. The metric was found to be more realistic for validating the estimated pose for a given manipulation task compared to prior art.

The final contributions are related to HRC which is a part of the next big industrial revolution, called *Industry 4.0*. The shift means breaking the existing safety practices in industrial manufacturing, i.e. removing the safety fences around the

robot and bringing the human operator to work in close proximity of the robot. This requires novel safety solutions that can prevent collisions between the co-workers while still allowing flexible collaboration. To address the requirements, a safety model for HRC is proposed and experimentally evaluated on two different assembly tasks. The results verify the potential of human-robot teams to be more efficient solution for industrial manufacturing than the current working methods. As a final study, usefulness and readiness level of augmented reality-based (AR-based) techniques as an user-interface medium in manufacturing tasks is evaluated. The results indicate that AR-based interaction can support and instruct the operator, making him feel more comfortable and productive during the complex manufacturing tasks.

CONTENTS

1 Introduction 15
1.1 Background and motivation 15
1.2 Publications and main results of the thesis 18
1.3 Outline of the thesis 20

2 Feature-Based Object Class Matching 21
2.1 Introduction 21
2.2 Background 23
2.3 Performance measures 24
2.3.1 Detector repeatability 24
2.3.2 Descriptor matching score 25
2.3.3 Coverage-N performance 25
2.4 Data 25
2.4.1 Image datasets 25
2.4.2 Ground truth annotations 27
2.5 Comparing detectors 28
2.5.1 Feature detectors 28
2.5.2 Evaluation 28
2.5.3 Results 29
2.6 Comparing descriptors 30
2.6.1 Feature descriptors 30
2.6.2 Evaluation 31
2.6.3 Results 31

2.7 Advanced analysis 32
2.8 Summary 38

3 Correspondence-Based 6D Object Pose Estimation 41
3.1 Introduction 41
3.1.1 Pose estimation methods 42
3.1.1.1 Template matching 42
3.1.1.2 Handcrafted features 43
3.1.1.3 Learning-based methods 44
3.1.2 Decomposition of the problem 46
3.2 Representing vision data as 3D 47
3.2.1 Pinhole camera model 47
3.2.2 Inverse model 49
3.2.3 From depth maps to point clouds 50
3.3 Point cloud simplification 50
3.3.1 Curvature filtering 52
3.3.2 Region pruning 53
3.3.3 Image datasets 54
3.3.4 Experimental setup 55
3.3.5 Results 56
3.3.6 Further analysis 57
3.4 3D local feature detectors and descriptors 60
3.4.1 Detectors 61
3.4.2 Descriptors 61
3.5 Matching 63
3.6 Correspondence filtering 64
3.6.1 Baseline methods 65
3.6.2 State-of-the-art 65
3.7 Estimating the pose from correspondences 68
3.8 Pose Estimation Metric for Robotic Manipulation 69

3.8.1 Background . . . 69
3.8.2 Probability of completing a programmed task . . . 72
3.8.3 Sampling the pose space . . . 74
3.8.4 Performance indicator . . . 79
3.8.5 Model validation . . . 80
3.9 Summary . . . 80

4 Safe HRC in Industrial Manufacturing . . . 85
4.1 Introduction . . . 85
4.1.1 HRC in manufacturing . . . 86
4.1.2 Collaborative robots . . . 87
4.2 Safe HRC . . . 89
4.2.1 Safety standards and criteria . . . 89
4.2.2 Safety strategies . . . 91
4.2.3 Vision-based safety systems . . . 92
4.3 Safety through robot control . . . 93
4.3.1 Speed and separation monitoring . . . 94
4.3.2 Potential field methods . . . 97
4.4 AR-based operator support system . . . 100
4.5 Summary . . . 102

5 Application of Safe HRC . . . 103
5.1 Introduction . . . 103
5.2 Shared workspace model . . . 103
5.2.1 HRC Zones . . . 104
5.2.2 Safety monitoring . . . 107
5.3 Setup . . . 108
5.3.1 Robot platform . . . 108
5.3.2 AR-based UI . . . 109
5.4 Experiments . . . 111
5.4.1 Task . . . 111

5.4.2 Methods 112
5.4.3 Performance metrics 114
5.5 Results 114
5.6 Summary 116
6 Conclusion 119
References 123
Publication I 139
Publication II 165
Publication III 175
Publication IV 181
Publication V 191
Publication VI 203

ABBREVIATIONS

kNN	k-Nearest Neighbour
2D	2-Dimensional
3D	3-Dimensional
ADC	Average Distance of Corresponding Points
AR	Augmented Reality
BoW	Bag of Words
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CNN	Convolutional Neural Network
DLP	Digital Light Processing
DoF	Degrees of Freedom
EVD	Eigenvalue Decomposition
GC	Geometric Consistency
GMM	Gaussian Mixture Models
GMR	Gaussian Mixture Regression
HG	Hough Grouping or Hand-Guiding Operation
HMD	Head-Mounted Display
HMM	Hidden Markow Model
HoG	Histogram of Oriented Gradients
HRC	Human-Robot Collaboration
ICP	Iterated Closest Point

LWR	Lightweigh Robots
MSE	Mean Squared Error
ORB	Oriented BRIEF
PCL	Point Cloud Library
PDF	Probability Density Function
PF	Power and Force Limiting
PFH	Point Feature Histogram
RANSAC	Random Sampling Consensus
SHOT	Signature of Histograms of Orientations
SI	Search of Inliers
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SMS	Safety-rated Monitored Stop
SSM	Speed and Separation Monitoring
SURF	Speeded-Up Robust Features
SVD	Singular Value Decomposition
ToF	Time-of-Flight
TRE	Translation and Rotational Error
UI	User-Interface
VOC	Visual Object Categorization

LIST OF ORIGINAL PUBLICATIONS

This thesis is based on the following articles, which are referred to in the text by notation [**P1**], [**P2**], and so forth.

P1 A. Hietanen, J. Lankinen, J.-K Kämäräinen, A. G. Buch, and N. Krüger. "A comparison of feature detectors and descriptors for object class matching. "*Neurocomputing* 184, pp. 3-12, 2016.

P2 A. Hietanen, R.-J. Halme, A. G. Buch, J. Latokartano, and J.-K. Kämäräinen. "Robustifying correspondence based 6D object pose estimation. "*International Conference on Robotics and Automation (ICRA)*, pp. 739-745, 2017.

P3 A. Hietanen, R.-J. Halme, J. Latokartano, R. Pieters, M. Lanz, and J.-K. Kämäräinen. "Depth-sensor-projector safety model for human-robot collaboration. "*International Conference on Intelligent Robots and Systems (IROS) Workshop on Robotic Co-workers 4.0*, 2018.

P4 A. Hietanen, A. Changizi, M. Lanz, J.-K. Kämäräinen, P. Ganguly, R. Pieters and J. Latokartano "Proof of concept of a projection-based safety system for human-robot collaborative engine assembly. "*International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1-7, 2019.

P5 A. Hietanen, R. Pieters, M. Lanz, J. Latokartano, and J.-K. Kämäräinen. "AR-based interaction for human-robot collaborative manufacturing. "*Robotics and Computer-Integrated Manufacturing (RCIM)*, 2020

P6 A. Hietanen, R. Pieters, M. Lanz, J. Latokartano, and J.-K. Kämäräinen. "Object Pose Estimation in Robotics Revisited" *arXiv:1906.02783*, 2020

Author's contribution

Antti Hietanen is the main author of all the publications in this thesis. Joni Kämäräinen was the main supervisor for all the publications, in terms of discussing ideas, and giving feedback on publication writing and experiments.

The main idea of publications **[P1-P6]** was discussed together with Joni Kämäräinen. The rest of the co-authors guided in the usage of laboratory equipment, help with construction of the experimental setups and/or gave valuable comments during the writing process. In addition, Minna Lanz kindly provided the robot hardware and laboratory facilities for the robotic research conducted in **[P2-P6]**. In all the publications, the software implementation and experiments have all been conducted by Antti Hietanen.

1 INTRODUCTION

1.1 Background and motivation

During the last decade, we have started to see a new generation of robots that are driven by advances in artificial intelligence and hardware technology. The robots have started to appear in completely new domains such as healthcare, education and even in our households for supporting in daily routines. In addition, due to the current requirements in automated industry, the traditional industrial robots have started to evolve from isolated work cells towards more flexible and autonomous agents that can apply dynamic strategies in complex and unpredictable environments. However, the robots' capabilities are still limited and more work is required to gain their full potential.

In computer vision, we are assigned to solve various task on visual input. Some examples are visualized in Fig. 1.1. The most generic one is *classification*, where the task is to classify an image according to its visual content i.e. in which class the object belongs. In contrast, *detection* is the task of localizing the object within an image and commonly includes estimating the object scale or the full 2D bounding box around the object. If the full image is given the detection system can provide location and instance information for multiple objects. Beyond 2D detection, we identify the object location in the 3D world, requiring the 3D position and 3D orientation of the object. Finally, *segmentation* is the task of assigning class labels to each pixel in the image, eventually giving perfect localization of the object. For instance in Fig. 1.1, the vision system has semantically classified all the pixels in the image to *person*, *snowboard* and *background*. In general, visual reasoning can be challenging task due to numerous reasons. For instance, the object class *car* might contain all the four-wheeled vehicles and such a large intra-class variation in appearance and shape makes the recognition task challenging. In addition, illumination changes or poor lightning conditions can change the appearance of an object and add undesired variability.

Figure 1.1 Different tasks in visual recognition.

In the majority of vision guided robotic applications, the main task consists of localizing known objects from the input image, which corresponds to the detection problem above (see Fig. 1.2). For instance in assembly lines or automated warehouses the robot has to accurately localize the object in order to grasp it successfully. Precise object localization is especially important in industrial applications, such as in welding and part installation, where the system has to comply with strict manufacturing tolerances. Most of the robots today are equipped with depth sensors which can calculate the distance of the scene objects respect to the sensor. The depth information can be further projected to 3D point clouds, providing important geometrical cues about the objects in the scene and enables detection of texture-less objects. In this case, the object detection can be accurately performed by estimated the 6D pose of the object i.e. giving 3D position and 3D orientation of the object. Working directly on 3D data has its advantages over 2D data as it is less affected of varying object appearance under different viewpoints and lightning conditions. However, the pose estimation can be severely affected by other means such as occlusion or due to undistinctive appearance of the object. For instance, the pose of a cup can be only detected uniquely if the handle of the cup is visible.

Another important application domain of computer vision and robotics is human-robot collaboration (HRC). HRC is part of the next big industrial revolution, the so called *Industry 4.0*, that combines new technology realms such as big data analytics, cyber physical systems and sensor networks in the hope of increased overall manufacturing value. In contrast to fully automated warehouses and factories, the primary target in HRC is not to replace the human worker but combine the strengths

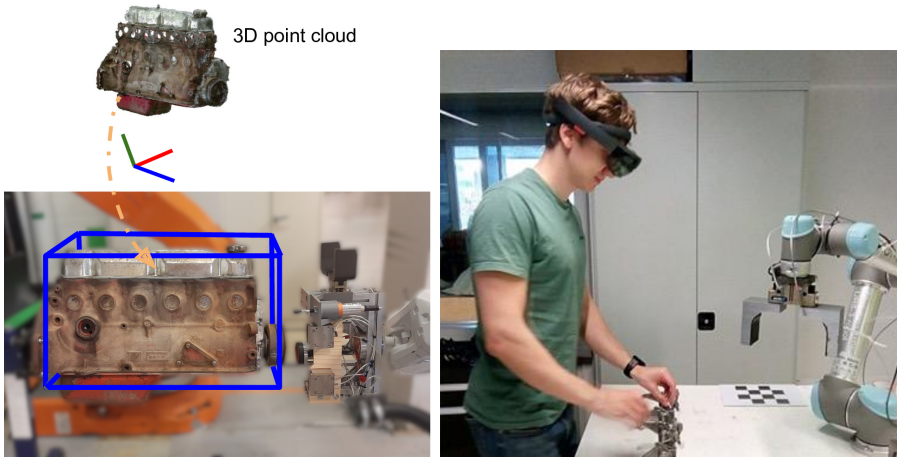


Figure 1.2 Visual recognition in robotic applications. The 6D pose of a motor engine is estimated for automotive disassembly (left). Head-mounted display instructs the human operator during a collaborative task with a light-weight robot (right).

of both worlds: repeatability and strength of a robot with the ability of a human to judge, react, and plan. The shift requires breaking the existing safety standards, which require the robot to work far away from humans in isolation. In HRC, the human and robot collaborate in close proximity, which creates big challenges from the safety point of view. Therefore, it is necessary to create novel safety approaches that are capable of detecting potential safety hazard while still allowing close interaction. In addition, seamless two-way communication channel between the human and robot is required for safe and efficient collaboration. In particularly, augmented reality (AR) has a high potential to be an effective medium to instruct the human operator in a complex task by augmenting the environment with virtual information. However, it is unclear how mature the AR-based technology is yet for real industrial manufacturing (see Fig. 1.2).

As already mentioned, the main objectives of this thesis is to study computer vision and its applications in robotics. In particular, the methods in question are divided into three distinct categories: 1) object class matching, 2) 6D pose estimation and 3) HRC. Regarding the categories, the main research questions which we consider in this thesis can be listed as follows:

Q1: "How well does the 2D local features perform in object class matching settings?"

Q2: "How can we robustify the existing model-based 6D pose estimation methods in scenarios, where the localized object has nondiscriminative surface structure?"

Q3: "How can we realistically measure the performance of an estimated object pose for a robotic manipulation task? "

Q4: "Can human-robot teams be more efficient solution than current working practices in industrial manufacturing?"

Q5: "What is the readiness level of the AR-based technology as an user-interface medium for manufacturing industry?"

1.2 Publications and main results of the thesis

The main results and the developed methods are published in one workshop paper [**P3**], two conference papers [**P4**, **P2**] and two journal articles [**P5**, **P1**]. In addition, one paper is currently under peer-review [**P6**]. The summary of the publications is the following:

Local feature detector and descriptor comparison for object class matching – [P1]

The first publication extends the well known 2D local feature detector and descriptor benchmark by Mikolajczyk et. al [95, 97] to class matching settings. In particular, we were interested to study how well the recent feature detectors and descriptors can find "common codes" between two object examples from the same class. For instance, a scooter and Harley-Davidson are both from the motorcycle class but there is a clear difference between the two in terms of shape and appearance. In contrast, one can still recognize semantically similar parts from the objects such a handlebar or pair of wheels. In the experiments, we evaluated the recent detectors and descriptors on multiple datasets using different performance metrics, including an alternative performance measure: *Coverage-N*. As the main results, the performance of detector-descriptor pairs on class matching settings is poor and specialized descriptors for visual class parts and regions are needed.

6D pose estimation for robotic manipulation – [P6, P2]

In the second publication [**P2**], the local features were extended to 6D pose estima-

tion where 3D-to-3D correspondences are used to fully localize the target object from the sensor input. Based on our earlier findings, repetitive or simple object geometry can significantly decrease the estimation accuracy and therefore two robustifying methods were proposed: *curvature filtering* and *region pruning*. The former method removes points from the object surface that are within low curvature areas. The latter processes the surface as local regions for which a good combination is sought by a trial-and-error procedure. Based on the experiments, the relatively simple algorithms were able to improve the accuracy of several pose estimation methods and were later utilized in a vision guided maintenance task where a tool of an autonomous ground vehicle was changed[1].

The publication [**P6**] proposes a completely new pose estimation evaluation metric for robotic manipulation. The previous works on the topic have mainly focused on metrics that rank the estimated poses solely based on the visual perspective i.e. how well two geometric surfaces are aligned. However, it is unclear how well the existing metrics can validate the estimated pose for a real robotic task. To address this, we propose a probabilistic evaluation metric that ranks an estimated object pose based on the conditional probability of completing a robotic task given this estimated pose. In addition, we present a procedure to generate automatically a large number of random grasp poses and corresponding task outcomes that are then used to estimate the grasp conditional probabilities[2]. In the experiments, the metric was found to be more realistic for measuring the estimated pose "goodness" for a given manipulation task compared to prior art. Together with our proposed evaluation metric we introduce a public benchmark containing an industry relevant RGB-D dataset with real automotive parts and approximate 600 test images.

Human-Robot Collaboration in industrial manufacturing – [P5, P4, P3]

In [**P3**], a safety model for HRC is proposed where the shared workspace is divided spatially to dynamic virtual zones, each having separate safety features. The zones are modeled and monitored by a single depth sensor overseeing the shared workspace. For interaction and feedback, a projector-camera user-interface was implemented. The proposed and a baseline safety system were experimentally evaluated in a simply assembly task[3]. In [**P4**], the previous work was extended to a

[1] https://youtu.be/U1GnHAlLaPE
[2] https://youtu.be/g4e_-p4fTEI
[3] https://youtu.be/CFKKANvWc3A

real diesel engine task and a work allocation schedule between human and robot resources was defined. In addition, the work introduced an important extension by extending the safety zones around the carried object since the assembly task included heavy and sharp objects. Our results from two different assembly tasks indicate the human-robot teams to be more productive than the existing work practices in industrial manufacturing without compromising the safety of the human co-worker. In the final publication [**P5**], usefulness and readiness level of two different AR-based devices, projector and Microsoft HoloLens, as an user-interface medium in manufacturing task, were evaluated. The qualitative and quantitative results from the experiments[4] indicate that projector-based interaction can support and increase the comfort of the human operator during the task while HoloLens was found surprisingly unpractical due to various reasons [**P5**].

1.3 Outline of the thesis

In Chapter 1, the motivation for the thesis and the content of each publication is summarized shortly. Chapter 2 introduces the local feature benchmark for class matching settings. The background related to the topic is briefly discussed and the main effort is in the explanation of the evaluation framework and main results from [**P1**]. In Chapter 3, a complete presentation of the 3D-to-3D correspondence-based pose estimation pipeline is given along with the contributions from [**P2**, **P6**]. Chapter 4 introduces HRC in industrial manufacturing and focuses on different safety techniques and strategies during the co-operation with a special attention to vision-based techniques, such as the one presented in [**P3**]. Chapter 5 focuses on publications [**P4**, **P5**] and describes the HRC safety model and its application in manufacturing industry. Finally in Chapter 6, the main achievements of the thesis are summarized. All six original publications [**P1**, **P2**, **P3**, **P4**, **P5**, **P6**] can be found at the end of the thesis.

[4] https://youtu.be/-WW0a-LEGLM

2 FEATURE-BASED OBJECT CLASS MATCHING

2.1 Introduction

Local feature detectors and descriptions have been the main building blocks of many computer vision algorithms during the past decades. They have been used successfully in many different applications, such as in wide baseline matching [133], object detection [96] and robot localization [120]. In wide baseline matching, one of the most typical task is 3D reconstruction, where the camera has to view the target object or scene from multiple viewpoints to cover all the aspects for accurate reconstruction. In such a scenario, local features are used for finding corresponding image points between two images of the sequence and the features have to tolerate significant rotation and translation of the camera between the views. In addition, the features have to cope with perspective change, blur and visual noise produced by the camera. Another interesting use of local features is object detection, where the main target is to estimate the location of the object in the input image. The task can be difficult for numerous reasons such as occlusion (the target object is partially hidden by other objects) or multiple instances of the same or similar objects in the scene. One of the main advantages of local features is that the whole object or scene is not required to be fully visible in order to successfully complete the recognition task.

A distinct application of feature-based matching is visual object classification where the problem is to classify an object to a general class such as *dog*, *car* or *bicycle*. This is a challenging problem as despite the fact that instances from the same category share similar physical properties, they are not exactly the same (see Fig. 2.1). The primary target is to identify and encode the same key characteristic that emerge between different objects from the same class. Bag-of-words (BoW) [123] and Histogram of

Oriented Gradients (HOG) [28] are common methods that utilize local features for object classification. BoW treats features as words and generates a codebook from a large number of words extracted from class examples. The generated codebook can be then used to create a frequency histogram of words of an image and compared against a histogram generated from an other image to measure the similarity between two images. The HOG feature calculates the histogram of gradient orientation in localized portions of an image and performs well with images having lots of edges and corners. In the original paper, Dalal et. al [28] used HOG features and trained Support Vector Machine (SVM) to detect pedestrians from images. Another interesting application of local features is unsupervised alignment of object class images [75]. The main objective is to learn visual object parts that can be reliable matched between different object instances from the same class. Typical use-case of unsupervised image alignment is to enhance the image annotation process, which is often done manually using expensive manpower.

Recently, several systems based on convolutional neural networks (CNN) have been successfully utilized for 2D classification, e.g. [73, 121]. Instead of using handcrafted features, the CNN is based on learned feature representation and can combine feature extraction and classification within one powerful architecture. The recent works have shown that the learned features can be invariant to extreme appearance variances, for instance between day and night [33, 148] and different weather conditions [118].

Figure 2.1 Examples of object instances from a single class (chair).

2.2 Background

The local feature detectors seek patterns from an image which have distinctive structure, such as edges, blobs or other small patches that differ from its immediate surroundings by texture, color, or intensity. Local features are computed from multiple locations in the image and as a result we get multiple feature vectors from a single image. These areas are then encoded to a vector representation using local feature descriptors and compared against descriptors extracted from another image by using simple distance metrics. The topic has gained lots of attention within the vision community and large number of different local feature detectors and descriptors have been presented. Comprehensive explanation of characteristics of different methods can been found in [80, 131]. Among the detectors, the most important property is the detector repeatability i.e. given two images of the same scene under different observing conditions, a high percentage of features should be extracted from parts of the scene that are visible on both of the images. The main objective of local feature descriptor is to encode the detected point or region *distinctively*, i.e. there is a low probability of matching the descriptor with a part of the object or scene that does not correspond to the same location in the other image. One of the most successful local feature is the Scale Invariant Feature Transform (SIFT) [85], which has been experimentally proven to be invariant against various transformations in the image domain. Today, there is wide variety of detector-descriptor pairs to choose from and typically one can narrow the choice based on the task requirements.

The standard way of evaluating the local feature detectors and descriptors has been already well established in [95, 97]. The works include reference test sets of images and evaluation metrics, on which future local feature detectors and descriptors can be fairly evaluated. The evaluation framework is mainly targeted for wide baseline matching and other applications in which we have images of the same scene. The evaluation framework evaluates the overlap of the detected areas of interest (detector test) as well as how well these regions actually match (descriptors test). The framework uses a small set of real images with variety of photometric and geometric transformation applied to them. The image set contains image pairs of scenes of distinctive edge boundaries (e.g. graffiti, building) and repeated texture of different forms (e.g. brick wall). For each image pair a ground truth plane projection transformation is provided for aligning the two images.

In this chapter we focus on the publication [**P1**] which extends the wide baseline benchmarks [95, 97] for local feature detectors and descriptors to the class matching setting. In the following, we evaluate the detectors and descriptors from publicly available repositories: OpenCV[1] (*cv*), VLFeat[2] (*vl*) and FeatureSpace[3] (*fs*). The test images are selected from three different databases and in addition to standard performance metrics, we investigate the effect of using multiple best matches ($K = 1, 2, ...$) and with an alternative performance measure: Coverage-N.

2.3 Performance measures

2.3.1 Detector repeatability

The main objective function of a feature detector is to achieve high repeatability and accuracy between two same objects i.e. they should return the same interest regions from both of the objects. In this work, the repeatability and accuracy is measured using the metric adopted from [97] with the exception that interest points detected outside the object area removed as shown in Fig. 2.3. The metric calculates the relative amount of overlap between detected regions in two different images using the homography matrix $\boldsymbol{H}$ relating the images. The two regions are counted as a correct match if the overlap error is less than a threshold value τ_{dt}

$$1 - \frac{A \cap (\boldsymbol{H}^T B \boldsymbol{H})}{A \cup (\boldsymbol{H}^T B \boldsymbol{H})} < \tau_{\mathrm{dt}} \ , \tag{2.1}$$

where A and B represents the detected elliptic regions. In addition, before calculating the overlap error the corresponding regions are normalized. This is done because the bigger the regions, the smaller the computed overlap error and vice versa. After finding all the correctly matched regions, the repeatability rate of a feature detector can be calculated as:

$$repeatability\ rate = \frac{\#\ correct\ matches}{min(\#\ regions\ in\ image\ A, \#\ regions\ in\ image\ B)} * 100 \tag{2.2}$$

[1] http://opencv.org/
[2] http://vlfeat.org
[3] http://featurespace.org

2.3.2 Descriptor matching score

As we stated earlier a good descriptor should be discriminative to match only correct regions and also it should be robust to some small appearance variations between the examples. In particularly, given regions A and B in the reference and target image, we want to know how well the corresponding feature vectors $\vec{f}_A$ and $\vec{f}_B$ match in the description space. Again we consider sets of image pairs on which the error is calculated. The computed regions are used as ground truth for the descriptor evaluation. We consider a region match to be correct if the overlap error (see Eq. 2.1) in the image covered by two corresponding regions is less than τ_{dc}. Each descriptor from the reference image is compared with each descriptor from the transformed one and the closest descriptor based on Euclidean distance is returned. We count the number of correct matches and finally measure the descriptor matching score for an image pair as ratio between the number of correct matches and the number of total matches.

2.3.3 Coverage-N performance

In our preliminary testing, we noticed that some of the descriptors were able to find correct matches in challenging settings in which other descriptors performed poorly. For that reason, we introduce an alternative performance measure in this work: Coverage-N. Coverage-N corresponds to the number of image pairs for which at least N descriptor matches have been found. It should be noted that the choice of the detector has impact on the measurement as it determines the spatial locations in the image where the descriptors are computed.

2.4 Data

2.4.1 Image datasets

Detectors and descriptors were evaluated on three different image databases: Caltech-101 [37], R-Caltech-101 [72] and ImageNet [30]. Examples of object instances from each dataset are shown in Fig. 2.2. Caltech-101 image database contains images and annotations for bounding boxes and outlines enclosing each object. We chose

Figure 2.2 Example images from Caltech-101 (top), R-Caltech-101 (middle) and ImageNet (bottom) datasets.

Caltech-101 because it is popular in papers related to object classification and contains rather easy images for benchmarking. We selected ten different classes from the database to get a good view of the performance over different content: *watch*, *stop_sign*, *starfish*, *revolver*, *euphonium*, *dollar_bill*, *car_side*, *air_planes*, *motorbikes* and *faces_easy*. Every image was scaled not to exceed 300 pixels in width and height.

The Caltech 101 database however has some weaknesses: the objects are typically in a standard pose and scale in the middle of the images. To make our benchmark process more challenging we adopted the randomized version of the Caltech-101 database where we used the same classes but with varying random Google backgrounds where the objects have been translated, rotated and scaled randomly. Annotations for bounding boxes and outlines are provided.

To experiment with our detectors and descriptors on more recent images, we included the ImageNet dataset in our evaluation. ImageNet provides over 100,000 different meaningful concepts and millions of images. However, landmarks for bounding boxes and outlines for the objects were not provided and we had to mark them manually. Nine classes were selected for the experiments: *watch*, *sunflower*, *pistol*, *guitar*, *elephant*, *camera*, *boot*, *bird* and *aeroplane*.

2.4.2 Ground truth annotations

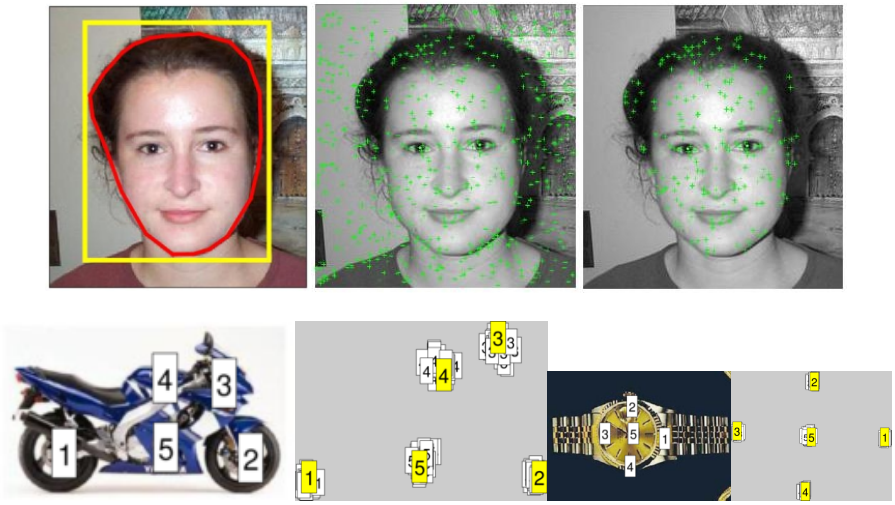


Figure 2.3 Top: bounding box (yellow line) and contour (red line) of the face, the detected SIFT features, and the remaining features after elimination. Bottom: landmark examples and multiple landmarks projected onto a single image (the yellow tags)

.

In our experiments, annotations for object bounding boxes and contour points are given for each image (see Fig. 2.3). Since we were only interested in measuring how well detected features found from the objects match within the same class, detected features outside the object contour were discarded. However, with more challenging randomized Caltech-101 dataset we only used the bounding boxes and some background features were detected.

From every image we manually selected 5-12 semantically similar landmarks which were then used to estimate the pair-wise image transformations using the direct linear transform [54] and linear interpolation. In Figure 2.3 is shown two object examples and the respective canonical image spaces, where all the annotated landmarks are projected.

2.5 Comparing detectors

2.5.1 Feature detectors

The detectors for the experiment were selected based on the early study [74] where the performance of nine publicly available detectors were evaluated. Among the top three detectors based on repeatability rate and number of correct matches we selected the Hessian detector (*fs_hesaff*) for our evaluation. In addition, we included in our preliminary testing four recently proposed and fast detectors: BRIEF [17], BRISK [79], ORB [112] and FREAK [2]. The best performance was obtained with ORB which we report in the results (*cv_orb*). Moreover, dense sampling (*vl_dense*) has replaced detectors in the top methods (Pascal VOC 2011 [35]) and as a fourth detector we also added SIFT (*vl_sift*) to our evaluation.

It is noteworthy that our evaluation differs from the earlier studies [74] in the sense that instead of using the default parameters for each detector we adjusted their meta-parameters to return the same number of regions for each image. This is justified as the work [101] claims that the number of interest points extracted from the test images is the single most influential parameter governing the performance. Indeed, as our results in the following sections show, the number of detected regions clearly has an impact on the detector performance. For ORB we adjusted the edge threshold, for Hessian-affine the feature density and the Hessian threshold, for SIFT the number of levels per octave, and for the dense the grid step size.

2.5.2 Evaluation

For the detector performance evaluation, the test protocol is similar to Mikolajczyk benchmark [97] which main points were discussed in Section 2.3.1. For each image pair, points from the first image are projected onto the second image by the homography transformation matrix estimated using the annotated landmarks. The interest points (regions) are described by 2D ellipses and if a transformed ellipse overlaps with an ellipse in the second image more than a selected threshold value a correct match is recorded. The reported performance numbers are the average number of corresponding regions between image pairs and the total number of detected regions. The detector performs well if the total number of detected regions is high and most of

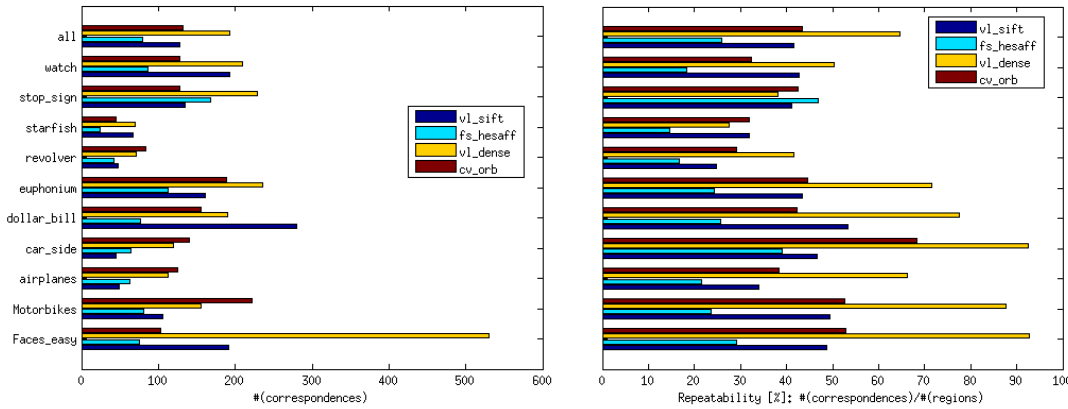
them overlap with the corresponding region on the second image. We adopt the parameter setting from [97]: a match is false if the overlap is less than 60% (i.e. $\tau_{dt} = 0.4$) and normalization of the ellipses to the radius of 30 pixels is used.

2.5.3 Results

Caltech-101 dataset. The results of the detector experiment on Caltech-101 dataset are reported in Fig. 2.4. Each detector was configured to return on average 300 regions. Based on the results, the *starfish* and the *revolver* categories were the hardest ones for all the detectors. Performance of dense sampling in *faces_easy* category is very good: it provides a lot of correspondence regions compared to other methods and the same regions are mostly found in both images.

With the adjusted meta-parameters the difference between the detectors is less significant than in the earlier evaluation [74] and the previous winner, Hessian-affine, is now the weakest. With the default parameters Hessian-affine returns almost five times more features than for instance SIFT, which made the evaluations too biased against the other detectors. The original SIFT detector performance without the parameter adjustment would be by order of magnitude worse. The new winner in the detector benchmark is clearly the dense sampling with a clear margin to the next best detector ORB. However, when computational time is crucial, the ORB detector seems tempting due to its speed.

Detecting more regions. In the above, we adjusted detector meta-parameters to return on average 300 regions for each image. That made detectors produce very similar results while using the default parameters in our previous work lead to completely different interpretation. It is interesting to study whether we can exploit meta-parameters further to increase the number of corresponding regions. We computed the detector repeatability rates as explained in Section 2.3.1 and the results are reported in Fig. 2.5. The figure also shows the number of returned regions by default parameters with the black dots. As expected, the results showed that the meta-parameters have almost no effect on the dense detection while Hessian-affine, ORB and especially SIFT clearly improve as the number of the regions increase (SIFT regions saturate to the same locations approximately at 600 detected regions).



(a) **(b)**

Detector	*Avg # of corr.*	*Avg. rep. rate*
vl_sift	127.5	41.6%
fs_hessaff	79.3	26.0%
cv_orb	132.0	43.5%
vl_dense	192.3	64.6%

(c)

Figure 2.4 Detector evaluation in object class matching. Meta-parameters were set to return on average 300 regions. (a) average number of corresponding regions, (b) repeatability rates, and (c) the overall results table.

2.6 Comparing descriptors

A good region descriptor for object matching should be discriminative to match only correct regions while tolerating small appearance variation between the examples. These are general requirements for feature extraction in computer vision and image processing. Compared to the original work [95] the descriptor matches in our work are expected to be weaker due to the increased appearance variation.

2.6.1 Feature descriptors

In the descriptor evaluation we used detector-descriptor pairs. It should be noted that available descriptors are not guaranteed to work well with different implementations

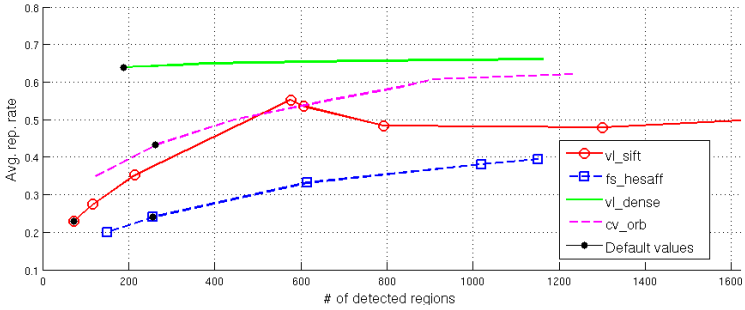


Figure 2.5 Detector repeatability as the function of the number of detected regions adjusted by the meta-parameters (defaults marked by black dots).

of detectors and thus we will use in our evaluation pair-wise detector-descriptor combinations only. From the earlier studies [74], we included the best performing pair: Hessian-affine and SIFT (*fs_hesaff + fs_sift*). Among the recent descriptors, we included the best performing detector (*cv_orb*) with two different descriptors: BRIEF (*cv_brief*) and SIFT (*cv_sift*). In addition, we report results for dense sampling and SIFT (*vl_dense+vs_sift*) and SIFT and SIFT (*vl_sift+vs_sift*). We also tested the RootSIFT descriptor from [4] that achieved better performance in their experiments, but in our case it provided insignificant difference to the original SIFT (mean: $3.9 \rightarrow 4.2$, median: $1 \rightarrow 1$).

2.6.2 Evaluation

We used the default ellipse overlap threshold 50% from [95] which is little bit looser than in detector evaluation, but also more strict thresholds were tested. The detectors meta-parameters were adjusted to return the same average number of regions (300). In the detector evaluation the mean and median numbers were almost the same, but here we report both since for the descriptors there was significant discrepancy between the values.

2.6.3 Results

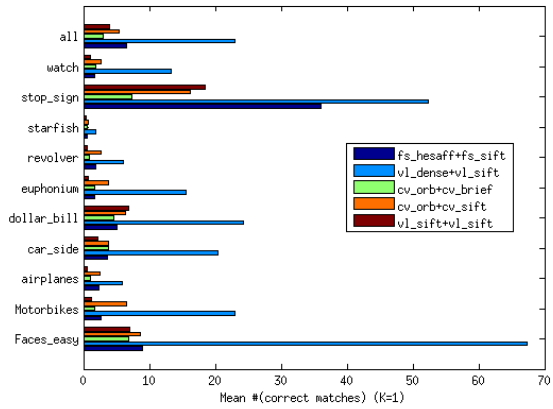
Caltech-101. The average and median number of matches for the descriptor evaluation are shown in Fig. 2.6. For many classes the matching performance is very

low, approximately 8 correct matches per image pair, and for instance the starfish category is extremely hard for every descriptor. However, the performance of dense sampling and SIFT is decent for most of the categories and superior compared to all other methods, achieving the average of 23.0% matches per class and median of 10.0% matches. The second best pair is Hessian-affine and SIFT and the rest of the methods are near behind with minor performance decrease. The more strict overlaps, 60% and 70%, provide almost the same numbers verifying that the matched regions do match well also spatially. In category wise the best results were obtained for the *stop_signs*, *dollar_bills* and *faces*, but the overall performance is poor. The best discriminative methods could still learn to detect these categories, but it is difficult to imagine naturally emerging "common codes" for other classes except the three easiest. It is surprising that the best detectors, Hessian-affine and dense sampling, were able to provide 79 and 192 repeatable regions on average, but only roughly 10% of these match in the descriptor space. Despite the fact that the SIFT detector performed well in the detector experiment, its regions do not match well in the descriptor space. The main conclusion is that the descriptors that are developed for wide baseline matching do not work well for different class examples.

Detecting more regions. As in Section 2.5.3, we studied the average number of matches as a function of the number of extracted regions. The result graph is shown in Fig. 2.7 and unlike the previous claim that the number of interest points is the most crucial parameter in feature matching [101] our results indicated that adding more regions by adjusting the detector meta-parameters provides only minor improvement to the average number of matches. Clearly, the best regions are provided first and dense sampling performs much better indicating that what is interesting for the detectors is not necessarily a good object part.

2.7 Advanced analysis

In this section, we address the open questions raised during the detector and descriptor comparisons in Section 2.5 and 2.6. The important questions are: why only a few matches are found between different class examples and what can be done to improve that? Why dense sampling outperforms all interest point detectors and does it have any drawbacks? Do our results generalize to other data sets.



Detector+descriptor	*Avg #*	*Med #*	*Avg # (60%)*	*(70%)*	*Comp. time (s.)*
vl_sift+vl_sift	3.9	1	2.8	1.6	0.15
fs_hessaff+fs_sift	6.5	2	5.9	4.9	0.22
vl_dense+vl_sift	23.0	10	22.3	20.2	0.76
cv_orb+cv_brief	3.0	1	2.9	2.7	0.11
cv_orb+cv_sift	5.4	2	4.8	4.1	0.37

Figure 2.6 Descriptor evaluation ($K = 1$ denotes the nearest neighbor matching, see Sec. 2.7 for more details). Top: average number of matches per class. Bottom: overall results table. The default overlap threshold is 50% [95], 60% and 70% results demonstrate the effect of the more strict overlaps. The computation times are average detector and descriptor computation times for one image pair.

ImageNet classes. To validate our results, we selected 10 different categories from the state-of-the-art object detection database: ImageNet [30]. The configuration set up was the same as in the section 2.6: the images were scaled to the same size as the Caltech-101 images, the foreground areas were annotated and the same overlap threshold values were tested. The overall results (see Fig. 2.8) indicated that the average number of matches is roughly half of the number of matches with Caltech-101 images which can be explained by the fact that the data set is more challenging due to 3D view point changes. However, the ranking of the methods is almost the same: dense sampling and SIFT is the best combination and the SIFT detector and descriptor pair is the worst. The results validate our findings with Caltech-101.

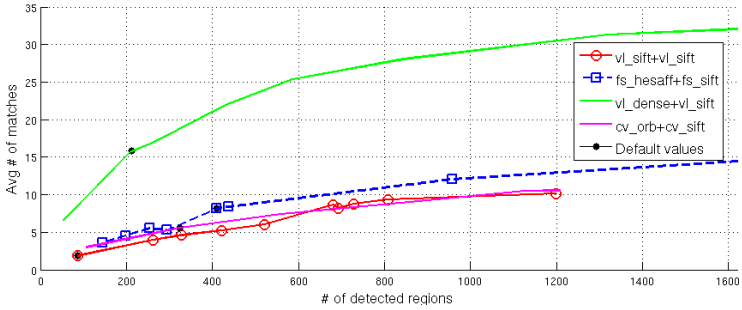
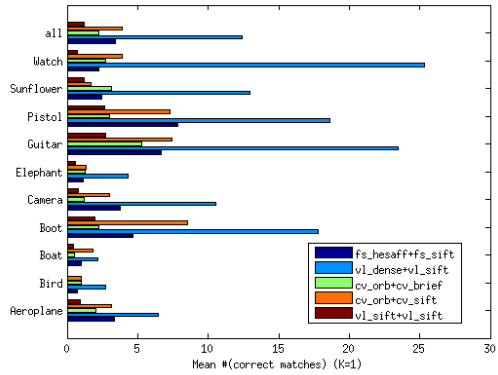


Figure 2.7 Descriptors' matches as functions of the number of detected regions controlled by the meta-parameters (default values denoted by black dots).

Beyond the single best match. In object matching, assigning each descriptor to several best matches, *soft assignment* [1, 20, 132], provides improvement and we wanted to experimentally verify this finding using our framework. The hypothesis is that the best matches in descriptor space are not always correct between two image pairs, and thus, not only the best, but a few best matches can be used. This was tested by counting a match as correct if it was within the K best matches and the overlap error was under the threshold. To measure the effect of multiple assignments, we used the Coverage-N measure (see 2.3.3 for more details). The coverage for $K = 1, 5, 10$ are shown in Figure 2.9 and Table 2.1. Obviously, more image pairs contain at least five ($N = 5$) than ten matches. Again, the configuration setup was the same as previously. With $K = 1$ (only the best match) the best method, VLFeat dense SIFT, finds at least $N = 5$ matches in 16 out of 25 image pairs and 13 for $N = 10$. When the number of best matches is increased to $K = 5$, the same numbers are 19 and 18, respectively, showing clear improvement. Beyond $K = 5$ the positive effect diminishes and also the difference between the methods is less significant.

Different implementations of the dense SIFT. During the course of work, we noticed that different implementations of the same method provided slightly different results. Since there are two popular implementations of dense sampling with the SIFT descriptor, OpenCV and VLFeat (two options: slow and fast), we compared them. The experimental evaluation showed slight differences between the different implementations, but the overall performances was almost equal, see Fig. 2.10. However, the computation time of the VLFeat implementation is much smaller com-



(a)

Detector+descriptor	*Avg #*	*Med #*	*Avg # (60%)*	*(70%)*
vl_sift+vl_sift	1.2	0	0.7	0.3
fs_hessaff+fs_sift	3.4	2	2.8	1.9
vl_dense+vl_sift	12.4	7	11.6	10.2
cv_orb+cv_brief	2.2	1	1.9	1.5
cv_orb+cv_sift	3.9	2	3.3	2.5

(b)

Figure 2.8 Descriptor evaluation with the ImageNet classes to verify results in Fig. 2.6.

pared to the OpenCV. In addition, the VLFeat fast version is roughly six times faster than the slower version of SIFT.

Randomized Caltech-101. With dense sampling the main concern is its robustness to changes in scale and, in particular, orientation, since these are not estimated similar to interest point detection methods. Therefore, we replicated the previous experiments with dense sampling implementations from VLFeat and OpenCV and the best interest point detection methods, Hessian-affine and SIFT, using the randomized version of the Caltech-101 data set. An exception to the previous experiments was that we discarded features outside the bounding boxes instead of using the more detailed object contour. The detector and descriptor results of this experiment are reported in Fig. 2.11. Based on the results, the detectors' performance were almost equivalent with the ones obtained using the Caltech-101 dataset. The comparison on

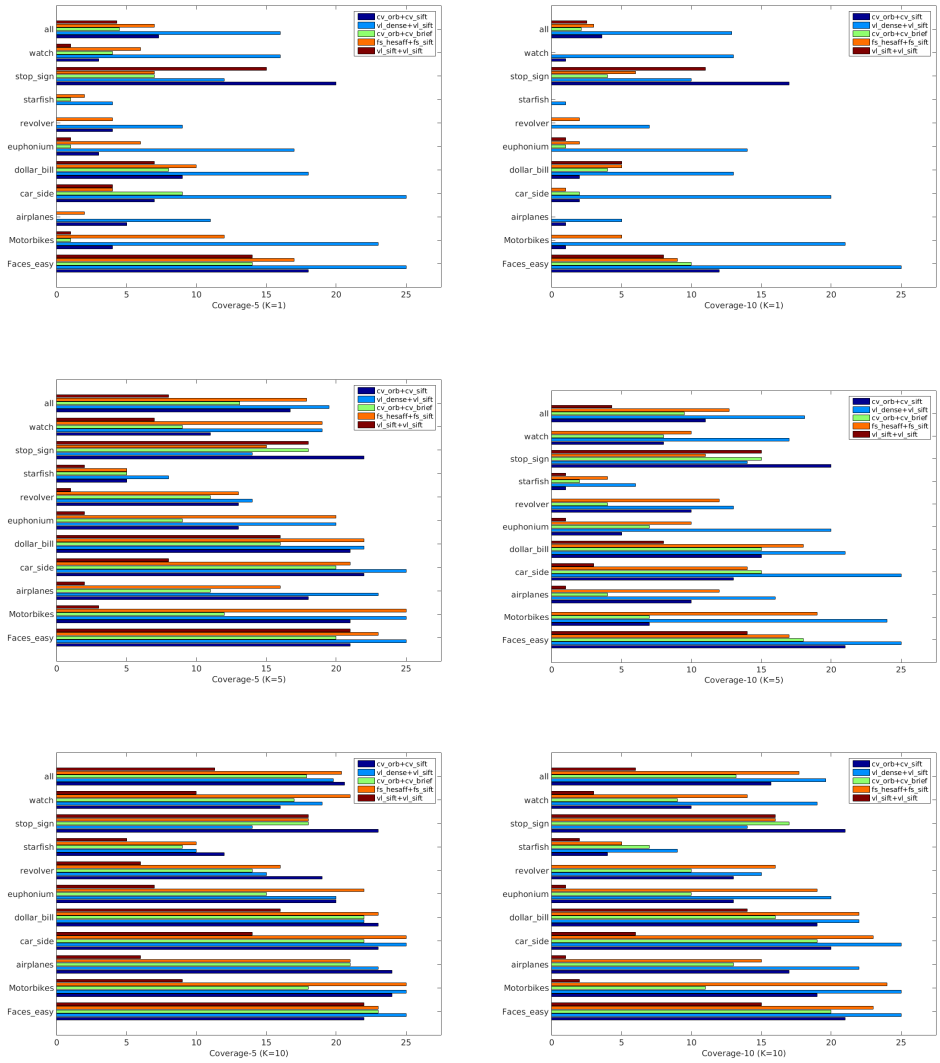


Figure 2.9 Number of image pairs for which at least $N = 5, 10$ (left column, right column) descriptor matches were found (Coverage-N). $K = 1, 5, 10$ denotes the number of best matches (nearest neighbors) counted in matching (top-down).

Table 2.1 Average number of image pairs for which $N = 5, 10$ matches were found using $K = 1, 5, 10$ nearest neighbors.

	Coverage-(N = 5)			*Coverage-(N = 10)*		
Detector+descriptor	*K=1*	*K=5*	*K=10*	*K=1*	*K=5*	*K=10*
cv_orb+cv_sift	7.9	16.7	23.0	3.6	11.1	15.7
vl_dense+vl_sift	16.0	19.5	19.8	12.9	18.1	19.6
cv_orb+cv_brief	4.5	13.3	17.9	2.1	9.5	13.2
fs_hesaff+fs_sift	7.3	17.9	20.4	3.5	12.7	17.7
vl_sift+vl_sift	4.3	8.0	11.3	2.5	4.3	6.0

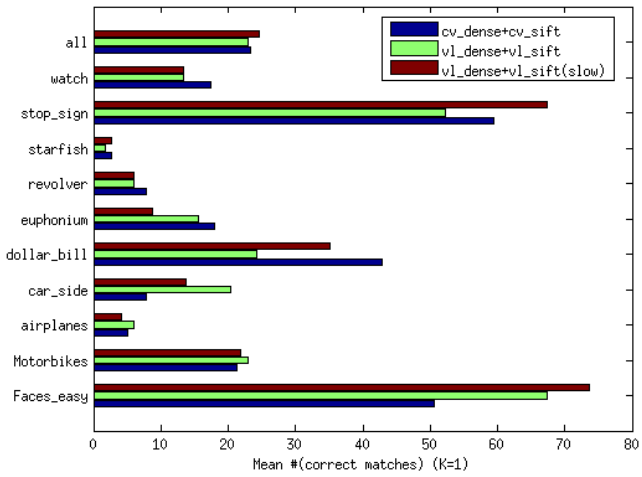


Figure 2.10 OpenCV dense SIFT vs. VLFeat dense SIFT (fast and slow) comparison.

detector-descriptors pairs showed that artificial rotations affects the dense descriptors and the performance was decreased by 35.6% — 44.3%. However, the detector-descriptor pairs with interest point detector were almost unaffected. It is noteworthy that the generated pose changes in R-Caltech-101 are rather small ($[-20°, +20°]$) and the performance drop could be more dramatic with larger variation. An intriguing research direction is detection of scaling and rotation invariant dense interest points.

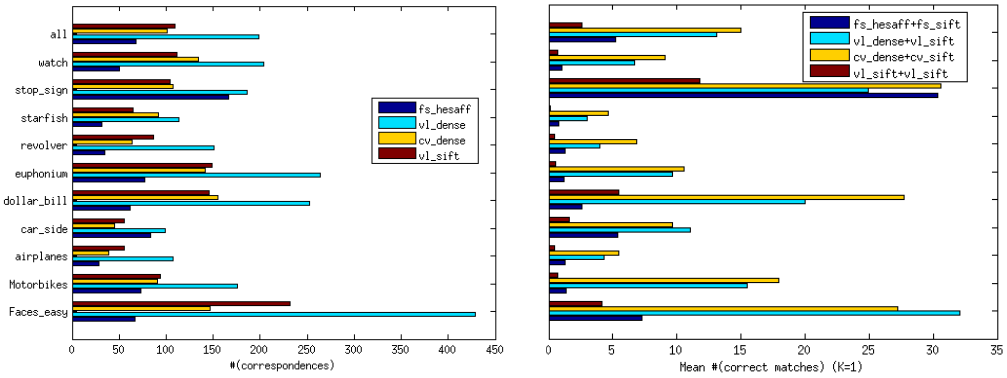


Figure 2.11 R-Caltech-101: detector (left) and descriptor (right). The detector results are almost equivalent to Fig. 2.4. In the descriptor benchmark (cf. with Fig. 2.6) the Hessian-affine performs better (mean: $3.4 \rightarrow 5.2$) while both dense implementations, VLFeat ($23.0 \rightarrow 13.1$) and OpenCV ($23.3 \rightarrow 15.0$) are severely affected.

2.8 Summary

In this chapter, the well accepted and highly cited interest point detector and descriptor performance measures by Mikolajczyk et. al [95, 97], the repeatability and number of matches, were extended to class matching settings with visual object categories. The recent and popular state-of-the-art detectors and descriptors were evaluated on various experiments using the Caltech-101, R-Caltech-101 and ImageNet datasets.

With our proposed framework we identified that dense sampling outperforms interest point detectors with a clear margin. It is the most reliable in the terms of repeatability rate and it also has the highest number of correspondences between image pairs. One of the most interesting findings was the number of detected features' relationship to the detection performance. The earlier winner Hessian-affine was surprisingly the weakest detector because of the adjustment of meta-parameters. The descriptor experiment showed that the original SIFT is the best descriptor including the recent fast descriptors. The descriptor experiment also showed that the choice of the detector which will be paired with the descriptor has a large impact to the results.

Generally, the detectors performed well, but descriptors' ability to match parts over visual class examples collapse. Also it is noteworthy to say that despite the fact that dense sampling performed well in the general evaluations, the method is fragile to object pose variation, while the Hessian-affine is the most robust against pose variations. Finally, using multiple, even a few, best matches instead of the single best match provides significant performance boost.

3 CORRESPONDENCE-BASED 6D OBJECT POSE ESTIMATION

3.1 Introduction

6D object pose estimation is an important problem in the realm of computer vision that determines the 3D position and 3D orientation of an object relative to a camera or based on some other known location in the environment. Estimating the pose of an object is usually considered the most challenging step in the object detection process where the target object has to be fully recovered in the sensor input. The research on 6D pose estimation has a long history and today it is a common task in many technological areas such as robotics, augmented reality and medicine.

In robotics there are two main applications for object pose estimation, namely object manipulation and navigation. In navigation the main target is to use a vision sensor to localize the robot within a known environment. Typical scenarios are patrolling, rescue operation and package delivery, in which an unmanned vehicle has to smoothly and safely navigate through the cluttered environment. In robotic based manipulation the fundamental property is to interact with objects in the environment, such as grasp and move it to a new location and finally install the object on correct position on the target object. Succeeding in such a task requires accurate 3D position and 3D orientation of the object of interest, i.e. 6D pose of the object. Especially objects with a complex shape might have only certain points on the surface where it can be reliably grasped by an end effector. More importantly, in industrial assembly the robotic task is commonly programmed based on a specific grasp pose with respect to the work part which is selected by an experienced engineer. Deviating from this pose will compromise rest of the operation, including moving the work part in the environment and installation of the object. In this case, the pose of the object has to be estimated precisely.

3.1.1 Pose estimation methods

In order to automatically handle various items by robots, accurate object detection and 6D pose estimation is required. In this section, the existing methods for estimating the 6D pose of an object are briefly reviewed and the methods are divided into three different research directions: *template matching*, *handcrafted features* and *learning-based methods*.

3.1.1.1 Template matching

Template-based matching is one of the earliest approaches for localizing the target object from the scene image. The matching works by sliding rectangular windows of several different sizes over the input image with predefined step size searching for the best candidate of the target object location. In practice it is not unusual to have thousands of different templates featuring various types of object characteristics for matching. During run-time each of these templates are exhaustively ran over the input image to capture appearance variations, which usually leads to a poor time complexity.

The first successful approaches based on template matching were proposed in the 1990s. The whole appearance of a target object from various viewpoints were used as model templates and the matching between models and inputs was done based on line features [76], edges and silhouettes [41], and shock graphs and curves [27]. However, most of the methods are very sensitive to illumination changes, artifacts and blur. For instance, the increasing amount of occlusion and blur is directly proportional to the number of extracted edges and curves which naturally has negative effect on the performance. In more recent works [55, 100] the authors do not use object boundaries but instead rely on image gradients. The templates from different view points and scene image are described using local dominant gradient orientations, which have shown to give good time complexity without sacrificing too much recognition performance. However, both of the methods are sensitive to background clutter, which can produce strong gradients disturbing the recognition pipeline. This is especially problematic if the interference is happening near the target object silhouette, which provides import feature cues for the method when dealing with texture-less objects.

Today the most often used baseline is the LINEMOD method proposed by Hinterstoisser et al. [56]. The method represent input objects using two feature modalities: orientation of intensity gradients and 3D surface normals.The input for the method is a RGB-D image, i.e. a registered color and depth image. The feature templates are generated automatically from 3D CAM models to reduce time and effort. After computing a similarity score for each of the templates, the ones having the highest score are retrieved and verified using consistency checks. Finally the best pose estimate provided by the template detection is refined using the Iterative Closest Point algorithm (ICP) [23]. Recently, a lot of work has been devoted to accelerate template matching, for instance by using hash tables [69] and GPU-optimized feature vectors [18].

3.1.1.2 Handcrafted features

Methods based on handcrafted features or simply features have a long history in 3D detection and have recently gained a lot of positive momentum due to the introduction of inexpensive RGB-D sensors capable of real-time 3D modeling. Compared to template matching, they are more robust against clutter and occlusions. The methods are commonly divided into two different groups: global and local methods.

Global methods describe the whole object model using a single or a small set of descriptors. One of the most promising methods was proposed by Drost et. al [32] which has gained reputation in a recent pose estimation benchmark [60]. The method creates a global description of the input point clouds based on oriented point pair features and matches them locally using a fast voting scheme. During training time a global description of the object model is created by pairing each of the model points to form a 4-dimensional point pair feature. All the point pairs are stored to a lookup table for faster indexing. During run time a set of reference points from the scene cloud is selected and all the remaining points are paired with reference points to create point pairs. Using the lookup table the point pair features are matched between the scene and global model description and a set of potential candidate matches is retrieved. Each of the candidates cast a vote for an object pose in a Hough-like voting scheme and finally the peaks in the accumulator space are extracted and used as the most prominent pose candidates. Due to its success a lot of methods have been proposed to improve performance and gain the full potential of the method [10, 25, 57, 136]. For instance, Hinterstoisser et. al [57] proposed

a robustified version to address the inefficiency and sensitivity to 3D background clutter and sensor noise of the original method. In addition, the performance was improved by smarter feature sampling and using a slightly different voting scheme in the matching stage.

The research on local methods started to be really popular at the beginning of 2000s and they are still widely used in a range of vision based applications, including pose estimation. In contrast to global methods, the local methods use each pixel or a robustly found set of key points to contribute to the detection output. One of the earliest 3D local descriptors was proposed by Johnson et. al [66] which created a spin image description of oriented 3D points. In contrast to 2D descriptors, the proposed descriptor was able to discriminate texture-less objects and was less affected by variations in the viewpoint and illumination. During the last decade many 3D local feature descriptors have been proposed, most notably Signature of Histogram of Orientations (SHOT) [128] and Point Feature Histogram (PFH) [115] which have achieved promising results in the recent benchmarks [24, 50, 52, 53]. The local methods are commonly coupled with robust and iterative sampling techniques, such as Random Sample Consensus (RANSAC) [38], which can search the most optimal alignment of two sets of 3D points.

3.1.1.3 Learning-based methods

Machine learning techniques have been utilized for learning feature representations that can discriminate the input image to foreground objects/background, object classes and 6D object poses. In general, learning can be categories into three different techniques: supervised, semi-supervised and unsupervised. Supervised methods require training sessions where training samples along with corresponding ground truth information are used to learn the model parameters. The methods might have millions of parameters which require sophisticated training algorithms and a number of examples images to work well. This is a clear disadvantage as compared to methods based on templates and handcrafted features where the model optimization is much more straightforward and can be done systematically enumerating all possible candidates. In contrast, the unsupervised learning is a technique where we do not explicitly tell the model what to do with the dataset. Instead, the model should be able to find unknown patterns from the data without any given training labels. As the name suggest, the semi-supervised learning refers to techniques where the model

is trained using labeled and unlabeled data. The technique is particularly useful in situations where a small set of training samples is available but it would be too costly to label all the data. From now on we will focus mainly on supervised techniques.

In conventional learning approaches Latent-Class Hough Forests [125] have been used to recover 6D pose of an object. The author extends the traditional Hough Forest to perform one-class learning at the training stage and use at run-time iterative approach to infer latent class distributions. In [11] random forest based method encodes contextual information of the objects with simple depth and RGB pixels and as a final step RANSAC based optimization scheme is used to improve the confidence of a pose hypothesis. The method was later improved in [12] by auto-context algorithm to support pose estimation from RGB-only images and additional improvements to the RANSAC step.

Due to significant performance boost in the standard recognition challenge [73], the vision community started to pay attention to Convolutional Neural Networks (CNNs). One of the earliest work of using CNN to capture an object 6D pose was proposed by Wohlhart et. al [143]. The authors proposed a simple CNN model to learn a 3D descriptor which can be used for both object classification and pose estimation. The model is trained using RGB or RGB-D images of different viewpoints and by enforcing simple similarity and dissimilarity constraints between the descriptors. During run time k-Nearest Neighbor (k-NN) search with a simple distance metric is used to evaluate similarity between a database pose and a scene image. The method was evaluated on the dataset of Hinterstoisser et. al [56] and outperformed several other state-of-the-art methods in different configurations. However, instead of the full sized test images, only regions containing the objects to be detected were used during the evaluation. In [31, 68], the authors proposed an auto-encoder architecture that can learn deep representation of the target objects using random patches from RGB-D images. Kehl et. al [68] coupled the auto-encoder with codebooks whose entries represent local 6D pose votes sampled from different objects views. During the detection phase local patches from input images are matched against the codebooks and the matches having the highest score will cast a 6D vote for pose sampling. An another successful method by Tekin et. al [126] extended the single shot architectures [109] for 6D detection tasks by predicting the 2D projections of the corners of the 3D bounding box around the objects. The author claims to produce accurate 6D pose estimates without any additional post-processing and the algorithm runs in

real-time.

3.1.2 Decomposition of the problem

During the rest of the chapter, we are focusing on correspondence-based pose estimation and use point clouds as our input data. The target is to identify 3D keypoints on the object surface and encode the local geometry around a keypoint into a feature description that can be uniquely matched. The descriptors provide reliable object recognition, but for accurate pose estimation the best result can be achieved by registering model points to corresponding scene points. For this registration process, obtaining correct point correspondences becomes a crucial task. In particularly, we are interested in finding the geometric transformation $\hat{\boldsymbol{\Upsilon}}$, consisting of translation $\vec{t} \in \mathbb{R}^3$ and rotation $\boldsymbol{R} \in SO(3)$, that localizes a point cloud model $\mathscr{M} \subset \mathbb{R}^3$ in the sensor input $\mathscr{M}' \subset \mathbb{R}^3$ based on some penalty function $\epsilon(\cdot)$:

$$\hat{\boldsymbol{\Upsilon}} = \arg\min_{\boldsymbol{\Upsilon}} \epsilon(\boldsymbol{\Upsilon}, \mathscr{M}, \mathscr{M}') \tag{3.1}$$

In this work, we consider *rigid* objects i.e. we assume the objects of interest have no moving parts or they cannot be deformed. Finally, we only consider rigid object *instances*, i.e. we are looking for a specific object that is different from all other objects in terms of visual aspects such as shape, color and material.

The complete description of a correspondence-based pose estimation pipeline is illustrated in Fig. 3.1, where the contributions are highlighted as green. The pipeline is divided into two different phases, *offline* and *online*. During offline, 1) parameters inherent to different steps in the pipeline are tuned in a supervised manner, 2) object models are encoded and saved to database for faster inference during online phase and 3) a pose estimation method giving the best performance based on a specific evaluation metric is selected for the task. During the online phase, the input image from the sensor is encoded and matched against target objects from the database. Finally, using robust sampling techniques, the 6D pose of an object is retrieved. In the following sections, the most important processing steps of the pipeline are discussed in chronological order along with the contributions. In particular, the work conducted in the publications [**P2**, **P6**] are discussed in Section 3.3 and Section 3.8, respectively.

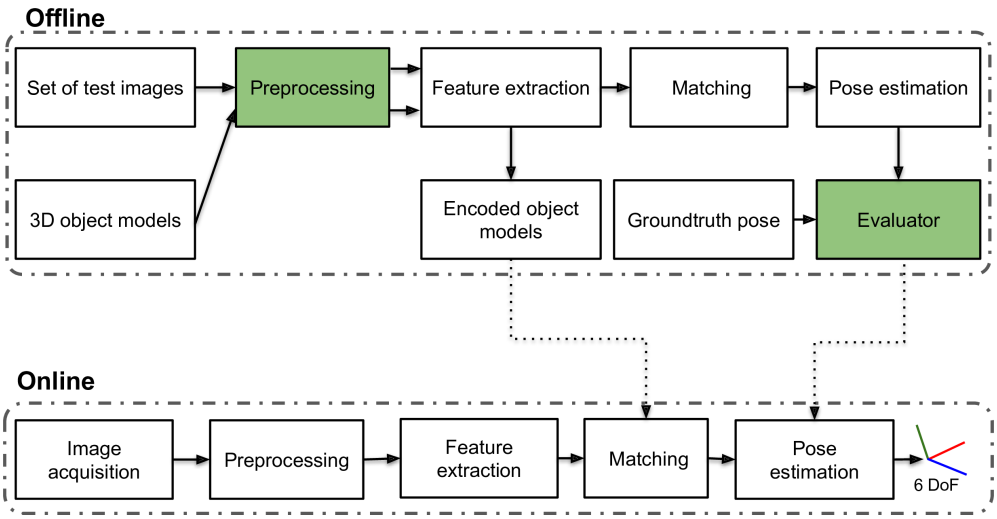


Figure 3.1 Various steps in a correspondence-based pose estimation pipeline.

3.2 Representing vision data as 3D

Autonomous robot agents perceive and process signals from the surrounding environment using vision sensors, such as RGB cameras, which capture and simplify the three-dimensional world into a 2D image plane. To understand the relationship between a 3D scene and its 2D projection onto a 2D camera plane, a mathematical camera model has to be described. Today the model is essential in many domains such as augmented reality, computer graphics and vision controlled robotics. The main purpose of the model is to create 2D projections of a 3D scene (*the forward-projection problem*) and to generate a 3D representation from a 2D image (*the inverse-projection problem*). Due to ever-increasing popularity of RGB-D sensors and rapid development of wearable augmented displays (e.g. Microsoft HoloLens) both of the problems are highly relevant.

3.2.1 Pinhole camera model

A simple camera can be abstracted by a pinhole camera model in which light rays travel a straight line from an object in the scene through a pinhole to a focal plane. The parameters of the pinhole camera model are encoded into a 4-by-3 matrix called

Figure 3.2 Sensor measurements from the Kinect v2 sensor. Left: Gray colored depth map encoding distance information from the surfaces of scene to the camera viewpoint. Foreground objects are colored darker and farther out are lighter. Middle: An RGB image aligned to the depth map. Some of the pixels are colored as black as they do not have corresponding pixel in the depth map. Right: resulting point cloud after inverse projection.

the *camera matrix*, which can be further decomposed into *intrinsic* and *extrinsic* matrices. The decomposition is useful as the intrinsic matrix describes solely the internal characteristics of the camera whereas the extrinsic matrix depicts the pose of the camera in the world. The camera matrix, $\boldsymbol{P}$, describes the position of a 3D world point on a image plane and it is defined by Hartley and Zisserman [54] as

$$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R}|\vec{t}] = \begin{bmatrix} f_x & 0 & x_o \\ 0 & f_y & y_o \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \end{array}\right] . \tag{3.2}$$

- The camera's extrinsic matrix $[\boldsymbol{R}|\vec{t}]$ describes the camera's location and orientation in the world. It consist of two components: a rotation matrix, $\boldsymbol{R} \in SO(3)$, and a translation vector $\vec{t} \in \mathbb{R}^3$. The vector $\vec{t}$ can be interpreted as the position of the world origin in camera coordinates, and the columns of $\boldsymbol{R}$ represent the directions of the world-axes in camera coordinates. In robotic applications, the world frame is commonly located in the robot base (e.g. center of the robot's mounting plate).
- The intrinsic matrix $\boldsymbol{K}$ transforms 3D camera coordinates to 2D homogeneous image coordinates. The focal length f_x and f_y measure the distance between the pinhole and the film (a.k.a. image plane) horizontally and vertically, respectively. In a true pinhole camera both f's have the same value but in practice they are different due to various reasons such as flaws in the camera sensor

or unintentional distortions caused by the camera's lens.

Now the coordinates of the projection point in pixels (u, v) can be created from a homogeneous 3D point $[x, y, z, 1]^T$ as

$$s \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{P} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} , \tag{3.3}$$

where s is the scaling factor.

Real life cameras produce geometric distortions to the captured image due to imperfect lenses. This is commonly observed as barrel (lines bend out of image center) or pincushion (lines bend towards the center) effect in the image. Commonly, the distorted point coordinates from Eq. 3.3 are corrected using a distortion function $\mathscr{D}$ that models the deviation of the real camera from an ideal perspective camera. The intrinsic matrix $\boldsymbol{K}$ and the distortion coefficients of $\mathscr{D}$ can be solved using camera calibration procedures [145].

3.2.2 Inverse model

The forward projection is the physical process where a camera captures a 3D point in the scene. However, in many vision aided applications the task is to invert the process and obtain the 3D point from 2D projection i.e. form the 3D representation of the scene as

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = s \cdot \boldsymbol{R}^T \boldsymbol{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} - \boldsymbol{R}^{-1} \vec{t} \ . \tag{3.4}$$

The scaling factor s is the unknown depth of the point. Because the forward projection maps all the 3D points along an optical ray to the same pixel location, the backward projection takes the 2D coordinate and returns a single point that has originated from the ray direction. Today the most used methods to measure depth is to use either a time-of-flight (ToF) or structured light sensor. Both technologies are used in many consumer grade RGB-D sensors which also provide color information

in addition to the depth map.

3.2.3 From depth maps to point clouds

Point clouds are a very useful representation of 3D geometry. Each point cloud represents a set of N 3D points $\{\vec{p}_i | i = 1,\ldots,N\}$ where $\vec{p} \in \mathbb{R}^3$. In many cases other attributes are associated to the points such as color $\vec{c} \in \mathbb{N}^3$ and normal $\vec{n} \in \mathbb{R}^3$ information. The depth map produced by a depth sensor can be directly converted to a point cloud using Eq. 3.4 if the sensor intrinsic parameters are known.

In many cases the obtained point cloud from a depth sensor is heavily simplified before processed further in the vision pipeline. For instance, the Kinect v2 depth sensor produces approximately 6.5 million points per a second (512×424 depth maps at 30 fps) and obviously creates requirements for memory capacity and computational power. In addition, the mapping will also produce various artifacts on the resulting point cloud due to imperfect measurements. For instance, the first version of the Kinect uses a structured light pattern for depth measurements and the light coding mechanism of the device causes large holes near the object boundaries and serious interference errors for multiple Kinects.

3.3 Point cloud simplification

The main target of different point cloud simplification approaches is to reduce the amount of data points in the point cloud and smooth out noise for better quality and approximation. Three popular and openly available filters in the Point Cloud Library (PCL)[1] are: *pass through filter*, *voxel grid filter* and *statistical outlier filter*. The pass through is the simplest of the approaches and a common step in every preprocessing pipeline. The filtering is based on two different operations. First, it removes non-finite point measurements. Second, it removes all the points along a specified coordinate axis that are further away than a predefined threshold value. For instance, a programmer can easily remove the background objects from the point cloud by simply adjusting the threshold value in the z-dimension (depth).

The pass through filter returns a sub-set of points from the original point cloud. The voxel grid method instead returns a point cloud with fewer number of points

[1] https://github.com/PointCloudLibrary/pcl

that should best represent the input point cloud as a whole. The voxel grid filter constructs a regular 3D grid over the entire input using a predefined resolution along each dimension. The set of points which lie inside a voxel are assigned to that voxel and will be approximated into one output point. In practice, two different approximations are used to represent the distribution of points within a voxel. In the first one, we take the average of the point coordinates inside a voxel and assigned that to be the voxel centroid. The second option is to simply select the point closest to the geometric center of the voxel. Clearly the latter option is more accurate but requires more computation power as the centroid point for each voxel has to be explicitly calculated.

The statistical outlier filter uses point neighborhood statistics to analyze whether or not a point in a point cloud is a outlier. The method is especially good at removing sparse outliers or so called *flying pixels* which appear close to object edges. During the first iteration the algorithm calculates the average distance for each point to its k nearest neighbors. By assuming that a spatial arrangement of a point neighborhood follows a Gaussian distribution, all the points whose neighborhood statistics are outside the global mean and standard deviation can be considered as outliers and removed from the point cloud.

In the following, we introduce two additional methods to simplify point cloud models, leading to more robust pose estimations from correspondence points [**P2**]. In our experiments, we have noticed that objects having a lot of repetitive structure or otherwise have really simple geometry can provide only a small amount of distinctively described local features. This will cause problems in the matching phase where the matches are found based on the feature similarities. The main idea of the proposed methods is to recognize points or patches on the object surface which constantly provide unreliable correspondence points and remove them from the point cloud. The methods should make the matching process invariant to pose ambiguities while still preserving the effectiveness against occlusion. The first methods is called *curvature filtering* that removes object model points within low curvature surfaces. The main assumption is that planar areas on the object model provide lots of false matches. The second method is called *region pruning* that divides the initial cloud to local regions based on a feature distance metric. The main idea is to use no assumption about which parts of the object model provide the best matches but use a trial-and-error procedure to find the best combination.

3.3.1 Curvature filtering

Curvature is a surface property that may affect to 3D object detection, tracking and pose estimation. For example, tracking does not converge on large planar areas where matches are equally good everywhere. On the other hand, sudden surface normal changes in high curvature areas, such as corners and edges, provide strong cues for tracking and pose estimation. There also exist a number of studies on perceptual experiments that demonstrate the importance of curvature in the human visual system [6, 9]. We compute the curvature value of a point as the *surface variation*

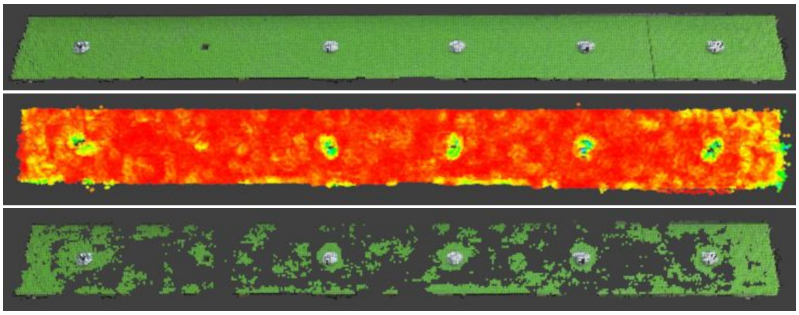


Figure 3.3 Point cloud model of the snow blade (top). The surface curvature of the snow blade shown as a heat map (middle) and robust sub-sets of the points after curvature filtering (bottom).

defined in [104]

$$\sigma = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \ , \tag{3.5}$$

where the λ:s are the eigenvalues (λ_0 being the largest) of the corresponding eigen vectors $\vec{v}_i$ of the covariance matrix $\boldsymbol{C}$

$$\boldsymbol{C} = \frac{1}{N_{curv}} \sum_{i=1}^{N_{curv}} (\vec{p}_i - \vec{\mu}) \cdot (\vec{p}_i - \vec{\mu})^T \ , \tag{3.6}$$

where N_{curv} is the number of points considered in the neighborhood of $\vec{p}_i$, and $\vec{\mu}$ represents the 3D centroid (mean) of the points. The number of neighbors N_{curv} and the curvature threshold τ_{curv} are the free parameters of the method. Points having lower curvature value than τ_{curv} will be removed from the point cloud (see Fig. 3.3 for an example).

3.3.2 Region pruning

The next method, region pruning, does not make assumptions about the shape properties around surface points, but divides the model point cloud into local regions by clustering. First we segment the model point cloud to supervoxels using the algorithm described in [102]. The grouping starts by dividing the 3D space of the model into a voxelized grid with resolution R_{seed}. Expansion of each cell is then done by local k-means clustering controlled by the feature distance measure

$$D = \sqrt{w_c D_c^2 + \frac{w_s D_s^2}{3R_{seed}} + w_n D_n^2} \ , \tag{3.7}$$

where D_s is the spatial distance by the seeding resolution, D_c is the Euclidean color distance in normalized RGB space, and the normal distance D_n measures the angle between surface normal vectors. Weights w_c, w_s and w_n control the influence of color, spatial and normal features respectively. Finally, we end up with N_{reg} regions (supervoxels) each having a central point $\vec{p}_n$. Now the main task is to find the combi-

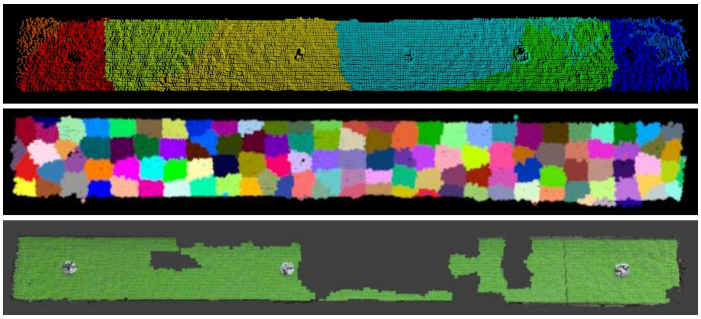


Figure 3.4 A snow blade divide into a supervoxel clusters: $N_{\text{reg}} = 10$ (top), $N_{\text{reg}} = 128$ (middle) and 72 out of 128 clusters after region pruning (bottom).

nation of the local regions that provides the best performance (see 3.4). This requires testing k-permutations of N_{reg} to find the best k regions out of the total number of regions N_{reg}. However, for a large k the exhaustive search of the best combination quickly becomes computationally intractable. Exhaustive search with k regions requires

$$\frac{N_{\text{reg}}!}{(N_{\text{reg}} - k)!k!} \tag{3.8}$$

experiments with all validation set scenes. The total number of tests is

$$\sum_{k=1}^{N_{\text{reg}}} \frac{N_{\text{reg}}!}{(N_{\text{reg}}-k)!k!} \tag{3.9}$$

combinations which is infeasible except for a very small N_{reg} ($10-15$).

3.3.3 Image datasets

Laser Scanner dataset. For the experiments, we included the well known Laser Scanner dataset[2], which has been used to evaluate 3D object recognition and 6D pose estimation methods [13, 92, 127]. From the dataset, we included four different target objects: *T-rex*, *Chicken*, *Parasaurolophus* and *Cheff*. The dataset contains in total 50 different test scenes captured using a high quality stereo camera system. In the test scenes, the target objects are placed in random order causing occlusions and clutter and in average 71%—77% of the points of the target objects are missing. The dataset contains also ground truth transformation matrices to align each model to the test scenes.

3D Tool Set dataset. The Laser Scanner dataset contains household objects with discriminative shape and size, which are not common in industrial environments or tasks. To include industry relevant objects, the 3D Tool Set dataset was created containing two different work tools: snow blade and container. The object models and test images were generated using our robot setup with the ABB IRB6640 manipulator and Kinect v2 sensor. The tools were attached on the robot end effector and then moved to different poses by the robot while the Kinect v2 sensor was capturing. For each tool, 20 images were generated with varying background and level of occlusion. 3D reconstructions of the tools were acquired manually by selecting multiple corresponding points from the different viewpoints and then merging all the views to a single image. The selected points were also used to generate the ground truth transformation matrices. All the artifacts and redundant parts were removed by the open-source mesh processing software MeshLab[3].

[2] http://staffhome.ecm.uwa.edu.au/~00053650/databases.html
[3] http://www.meshlab.net/

3.3.4 Experimental setup

Recognition pipeline. All the object models and test scenes are given as point clouds and processed as scene-model pairs. Input data is processed in the following manner: 1) First we downsample the model and scene using a voxel grid filter with a fixed resolution to limit the amount of data. The resolution was selected empirically to 1.0cm resulting in approximately $3,000 - 5,000$ points depending on the size of the input. 2) Model point cloud is altered by one of the robustifying methods, i.e. curvature filtering (curv) or region pruning (regp). 3) Next, a set of 3D local features are extracted from the model and scene surface. In our preliminary testing SHOT 3D descriptor performed best and was selected for the final experiments. 4) Finally the 3D features are matched between the model and scene and the pose is estimated using one of the following methods: *Search of Inliers (SI)* [13], *Geometric Consistency Grouping (GC)* [21] and *Hough Grouping (HG)* [127]. In the experiments we evaluated the pose estimation methods with default parameters, optimized parameters (opt) and using the robustifying methods (curv or regp). In addition, GC and HG were evaluated with and without RANSAC [38] as an additional robustifying method.

Performance measure. For assessing the pose estimation performance, we adopted the *average distance of corresponding model points* (ADC) performance metric [56], which measures the mean squared error (MSE) of all model points transformed by the estimated object pose and the ground truth object pose. Since some methods may completely fail for certain test scenes, we also reported best-50% and best-25% MSE values, which are less affected by estimation failures providing large errors.

Parameter optimization. The two important parameters for the curvature filtering are the number of neighbor points N_{curv} and the curvature threshold τ_{curv}. Based on the initial experiments we can make two observations: the neighborhood size must be large enough to compute a robust curvature estimate (≥ 5) and finding a suitable value for the curvature threshold is essential and is likely to depend on each model's properties.

One of the main parameter of the region pruning is the number of regions N_{reg} which also defines the computational complexity to find the best k combinations. It turns out that exhaustive search for all k-permutations is doable only for $N_{\text{reg}} \leq 10$,

but based on preliminary testing, for good results we typically need $N_{\text{curv}} \geq 100$. In our case, we selected $N_{\text{reg}} = 128$ for all the object models and randomly removed 10% of the regions and executed this random procedure 1,000 times. One should also note that although the robustifying methods do not significantly improve some of the methods, we can still remove insignificant points while maintaining the same or an even better pose estimation performance.

Each of the pose estimation methods had their own meta-parameters which had to be tuned for good performance. The parameters are commonly dataset specific and in our experiments the parameters were tuned individually for each object model. The optimization was done manually over specified subset of the parameter space and selected based on the ADC score. More details about the parameter optimization can be found in [**P2**].

3.3.5 Results

The results for the selected three methods and their variants are presented in Table 3.1 for the Laser Scanner dataset and in Table 3.2 for our 3D Tool Set. From the results we can make the following observations: GC approach provides the most accurate and robust pose estimation. For the Laser Scanner dataset objects GC variants are best for 10/12 cases and SI-opt (curv) wins 2/12. Based on the experiments, the performance of the pose estimation methods with default parameters was low and the default values are commonly far from the optimal ones; this is particularly evident for best-50% and best-25% MSEs, indicating that fewer poses are falsely detected (far from the true pose). Regarding the robustifying methods, there was no clear winner between curvature filtering and region pruning. The curvature-based filtering to robustify the methods does not improve HG and GC, but consistently improves SI making it comparable or even better than HG and GC. Region pruning consistently improves both GC and SI often achieving the best accuracy (8 out of 12 cases). On our own dataset the results are very similar, although the objects are very different from those in the Laser Scanner dataset - our objects contain many large planar areas which supposedly should benefit from curvature filtering. Again GC variant is the winning method in all cases (6/6). Clearly, the method of choice is GC with optimized parameters and curvature filtering as the GC-opt-RANSAC (curv) wins 4/6 cases. Based on the experiments, GC-opt-RANSAC (curv) was used

Table 3.1 Point cloud alignment errors for the Laser Scanner dataset. **Note:** *RANSAC is part of the method.

	Cheff			T-rex			Chicken			Parasaurolophus		
$\times 10^{-3}$	MSE	best-50%	best-25%	MSE	best-50%	best-25%	MSE	best-50%	best-25%	MSE	best-50%	best-25%
default parameters												
GC [21]	6.235	0.026	0.002	24.111	9.437	0.479	12.997	2.188	0.070	50.493	3.091	0.012
HG [127]	45.719	26.425	22.599	62.805	34.913	21.927	13.633	3.233	0.227	51.331	9.464	1.907
SI [13]	15.622	0.049	0.034	24.906	12.904	4.858	16.552	3.360	0.111	46.051	3.588	0.012
object optimized parameters												
GC-opt	5.108	**0.002**	**0.001**	**17.321**	8.015	0.197	11.175	1.727	0.042	46.253	2.697	0.009
HG-opt	7.586	0.520	0.003	17.557	12.496	7.334	10.592	2.829	0.227	46.772	8.679	0.624
SI-opt	5.300	0.021	0.010	27.700	11.600	3.900	13.000	1.678	0.012	46.000	3.503	0.005
optimized & RANSAC												
GC-opt-RANSAC	3.100	0.006	0.001	19.464	7.150	0.102	10.936	1.554	0.089	48.000	2.575	0.016
HG-opt-RANSAC	35.501	21.260	15.000	45.900	21.100	15.200	14.396	2.484	0.204	46.981	7.572	0.049
SI-opt*	5.300	0.021	0.010	27.700	11.600	3.900	13.000	1.678	0.012	46.000	3.503	0.005
optimized & curvature filtering												
GC-opt-RANSAC (curv)	6.900	0.036	0.010	21.440	13.838	7.779	10.537	1.581	0.100	45.600	2.720	0.024
HG-opt-RANSAC (curv)	13.930	5.207	2.937	21.881	9.711	3.198	12.374	1.989	0.179	50.663	3.852	0.141
SI-opt (curv)	3.900	0.017	0.007	21.900	8.385	0.384	10.017	1.252	**0.007**	45.900	2.963	**0.002**
optimized & region pruning												
GC-opt (regp)	**2.332**	0.004	**0.001**	18.326	**5.320**	**0.050**	**9.301**	**0.779**	0.016	**45.198**	**1.952**	0.006
HG-opt (regp)	14.670	9.718	9.134	32.136	21.249	15.897	29.902	15.622	11.512	60.179	22.303	16.045
SI-opt (regp)	4.600	0.018	0.007	22.600	8.300	1.300	16.734	3.496	0.120	46.695	3.131	0.082

in a case study demonstration where external recognition system supports a blade change during a maintenance task[4].

3.3.6 Further analysis

Figure 3.5 Five different object models from the 3D Motor part dataset.

On the previous datasets there was no clear winner between two robustifying

[4]https://youtu.be/U1GnHAlLaPE

Table 3.2 Method performance for the Outdoor Robot Tool dataset.

	Blade			Box		
$\times 10^{-3}$	MSE	best-50%	best-25%	MSE	best-50%	best-25%
default parameters						
GC [21]	6.2730	1.0320	0.1890	6.7880	2.6190	0.0002
HG [127]	6.0620	0.6960	0.1240	8.7000	5.4800	1.5330
SI [13]	2.4080	0.0200	0.0005	9.7780	4.9950	0.0389
object optimized parameters						
GC-opt	0.8671	**0.0003**	**0.0001**	5.7827	1.6394	**0.0001**
HG-opt	4.6077	0.2024	0.0510	6.7606	3.1600	0.0002
SI-opt	2.1690	0.0022	0.0005	6.3588	3.0081	0.0005
optimized & RANSAC						
GC-opt-RANSAC	0.4184	0.0004	0.0002	4.1384	0.0463	0.0002
HG-opt-RANSAC	0.7333	0.2010	0.0330	6.0916	1.7224	**0.0001**
SI-opt	2.1690	0.0022	0.0005	6.3588	3.0081	0.0005
optimized & curvature filtering						
GC-opt-RANSAC (curv)	**0.2280**	0.0004	0.0002	**2.9893**	**0.0113**	**0.0001**
HG-opt-RANSAC (curv)	0.2595	0.1288	0.0334	6.0283	0.0249	0.0002
SI-opt (curv)	2.1734	0.0020	0.0004	6.2161	1.4291	0.0004
optimized & region pruning						
GC (regp)	0.2744	**0.0003**	0.0002	5.1058	0.1475	0.0002
HG (regp)	2.2614	0.2367	0.0805	7.4540	2.8303	0.0006
SI-opt (regp)	2.2948	0.0014	0.0007	6.0578	2.0800	0.0014

methods. Due to that reason, we created a bigger dataset with separated training and test image sets for further analysis.

3D Motor part dataset. For the experiments, the 3D Motor Part dataset was created containing real objects from automotive industry: *motor block*, *compressor*, *rear axle*, *manifold block* and *exhaust pipe*. Similarly to the 3D Tool set dataset, the images of the motor parts were generated using a robotic setup. However, this time the data gathering process was done automatically using a modern industry robot ABB IRB4600. The robot with the Kinect v2 sensor was moved systematically around each of the objects capturing all the aspects and approximately 120 images were generated per-object. After the automatic data capture, one of the views was selected as the canonical view and all the other views were aligned with the canonical view auto-

matically using robot kinematics and robot-camera calibration. The reconstructed 3D objects were manually checked and refined using MeshLab. Similarly, the ground truth transformation matrices were generated automatically using the known transformation chain from sensor to object and finally the initial alignment was refined by the ICP[23] method.

Experimental setup. The evaluation protocol was copied from the previous experiments. As an additional method, we included a modified version [15] of the default RANSAC procedure into the comparison. In addition, GC and HG were evaluated without the additional RANSAC refinement step for fair comparison. The image datasets were divided into separated training and test set, with a 75% — 25% split. The method parameters were tuned on the training set and the ADC error of the optimized method on the test set was reported in the final results.

Results. The results for the 3D motor part dataset are shown in Table 3.3. The most striking finding is remarkable improvement of the GC [21] method with optimized parameters. This is important finding as the parameter optimization is straightforward to do and can be conducted similarly across different datasets. Moreover, without optimization the performance of GC is low and with optimized parameters it is the best performing method on average in our comparison. Without parameter optimization GC was the worst or second worst performing method for all five objects and using all error measures. With object specific optimized parameters GC becomes the best or second best for all objects and for 14 out of the 15 possible measures. The HG and RANSAC generally perform the worst with optimized parameters and remain clearly below GC and SI. While SI is clearly the best performing method with its default parameters it is inferior to GC with optimized parameters in all but one case (compressor object). Moreover, SI did not always benefit from optimization and GC systematically did for all objects and all error measures. The main finding of this experiment with the realistic industrial 3D motor part dataset is that the GC with object specific parameters is superior for 3D object pose estimation. On the other hand, unlike in our previous work with limited data the two robustifying approaches, curvature filtering and region pruning, do not provide systematic improvement for GC or any other method but in many cases lead to clearly inferior results. This is likely cause of overfitting with limited validation data.

Table 3.3 Point cloud alignment errors for the 3D motor part dataset.

	Motor			Compressor			Axis			Exhaust pipe			Exhaust pipe 2		
$\times 10^0$	MSE	best-50%	best-25%	MSE	best-50%	best-25%	MSE	best-50%	best-25%	MSE	best-50%	best-25%	MSE	best-50%	best-25%
default parameteters															
GC [21]	1.5377	0.6613	0.3376	0.9004	0.3686	0.1682	0.4181	0.1440	0.0834	0.8288	0.1197	0.0855	1.4777	0.4972	0.1639
HG [127]	1.1544	0.5171	0.3473	0.7426	0.4023	0.2774	0.3525	0.1370	0.0903	0.5336	0.1805	0.1290	1.0030	0.3397	0.2262
SI [13]	0.0467	0.0008	0.0005	**0.0131**	0.0008	0.0005	0.1226	0.0041	0.0015	0.1398	0.0010	0.0007	0.3272	0.0035	0.0004
RANSAC [38]	1.1324	0.2816	0.1039	0.2111	0.0175	0.0084	0.1638	0.0070	0.0032	**0.0464**	0.0049	0.0008	0.5586	0.0172	0.0023
object optimized parameters															
GC-opt	0.1495	0.0010	**0.0002**	0.0420	**0.0006**	0.0004	**0.0471**	**0.0020**	**0.0009**	0.1026	**0.0003**	**0.0002**	0.3312	**0.0009**	**0.0002**
HG-opt	0.1922	0.0123	0.0044	0.0335	0.0016	0.0006	0.1520	0.0243	0.0152	0.1089	0.0026	0.0010	0.4408	0.0143	0.0084
SI-opt	**0.0351**	0.0006	0.0004	0.0182	0.0009	0.0005	0.0630	0.0051	0.0019	0.1421	0.0009	0.0006	0.3221	0.0020	0.0005
RANSAC-opt	0.6625	0.4050	0.2881	0.1733	0.0160	0.0080	0.0503	0.0046	0.0028	0.0670	0.0032	0.0009	0.4480	0.0097	0.0024
optimized + curvature filtering															
GC-opt (curv)	0.0878	0.0010	**0.0002**	0.0806	0.0007	**0.0003**	0.0527	0.0023	0.0012	0.0848	**0.0003**	**0.0002**	0.3468	0.0010	0.0003
HG-opt (curv)	0.1367	0.0150	0.0072	0.0227	0.0015	0.0007	0.1354	0.0177	0.0124	0.1110	0.0022	0.0009	0.3860	0.0138	0.0076
SI-opt (curv)	0.0947	0.0007	0.0005	0.0957	0.0010	0.0006	0.1510	0.0045	0.0016	0.1765	0.0009	0.0006	0.3798	0.0050	0.0005
RANSAC-opt (curv)	0.6260	0.3765	0.2712	0.0709	0.0138	0.0053	0.0974	0.0033	0.0019	0.1179	0.0031	0.0012	0.5367	0.0124	0.0013
optimized + region pruning															
GC-opt (regp)	0.0551	**0.0004**	**0.0002**	0.1387	0.0906	0.0846	0.1480	0.0024	0.0014	0.1621	0.0004	**0.0002**	0.4741	0.0023	0.0006
HG-opt (regp)	0.1393	0.0103	0.0056	0.1291	0.0855	0.0783	0.1236	0.0122	0.0067	0.1037	0.0037	0.0018	0.3221	0.0128	0.0066
SI-opt (regp)	**0.0351**	0.0006	0.0004	0.0182	0.0009	0.0005	0.0630	0.0051	0.0019	0.1421	0.0009	0.0006	**0.3220**	0.0020	0.0005
RANSAC-opt (regp)	0.5863	0.3215	0.1747	0.2637	0.1023	0.0899	0.0397	0.0045	0.0022	0.1065	0.0038	0.0013	0.4517	0.0209	0.0020

3.4 3D local feature detectors and descriptors

After the input data has been preprocessed, the next step in the pipeline is to extract 3D local features from the object surfaces. The use of local features has been a broadly-used paradigm in detection systems during the past few years and methods such as SIFT [85] have been successful in various recognition tasks. Today, especially due the availability of low-cost 3D sensors, the robotic community consumes more and more 3D data, such point clouds, which are capable of providing 3D representation of the world. This has motivated the creation of new techniques and during recent years various 3D local feature detectors and descriptors have been proposed. The main design principles have been the same as in the 2D domain i.e. the features should be stable under common object transformation (translation, rotation and scaling) as well as geometric distortions produced by imperfect vision sensors. In particular, we want to generate feature representation $\mathscr{F} \subset \mathbb{R}^N$ of $\mathscr{M}$ and commonly feature vectors are calculated only for a subset of points i.e. $|\mathscr{F}| < |\mathscr{M}|$. A major limitation of 3D local feature-based methods is that the performance depends crucially on careful parameter tuning and the size of the local support region is one of the most important factors to consider [50]. In point cloud-based processing the locality is specified by the spherical support radius r. The considered surface point $\vec{p}$ in point cloud $\mathscr{M}$ is utilized using all the neighboring points $\vec{p}_i$ within the support $\mathscr{S}(\vec{p}) = \{\vec{p}_i \in \mathscr{M} : \|\vec{p} - \vec{p}_i\| \leq r\}$. Detailed description and evaluation of recent

methods can be found from [24, 50, 52, 53]. In the following, we will briefly go though the most common and publicly available 3D feature detectors and descriptors.

3.4.1 Detectors

The main goal of 3D feature detectors is to utilize the geometric characteristics of the underlying local surface and include the most distinctive regions for feature encoding and matching. A good detector should return distinctive and repeatable regions which can be effectively described and matched to prevent wrong point-to-point correspondences. However, there exist considerably less 3D than 2D keypoint detectors and commonly simple random or uniform sampling is used.

The vast majority of the existing 3D local feature detectors are adopted from the 2D domain. The four main 3D keypoint detectors implemented in the PCL library are Harris3D [122], NARF [124], Intrinsic Shape Signatures (ISS) [147] and SIFT3D [40]. The Harris3D keypoint detector is inspired from the traditional 2D Harris detector. Instead of using image gradients the method utilizes surface normal for corner detection and removes weak keypoints using non-maximal suppression. Similarly, the main ideas of the original SIFT are adopted to the 3D domain by using extended version of the Hessian matrix and introducing 3D sub-histograms. The NARF detector selects points based on the surface stability (to ensure a robust estimation of the normal) and where there are sufficient changes in the immediate vicinity of the query point. ISS measures the saliency of a query point $\vec{p}$ based on the Eigenvalue Decomposition (EVD) of the scatter matrix of the points belonging to the support of $\vec{p}$. The method is highly discriminative and especially design for 3D point clouds.

Recent evaluations has shown that the performance of feature matching can be improved when combined with 3D local feature detection methods (as opposed to uniform sampling or a random selection of the feature points) [24, 50]. However, the performance is improved only on certain datasets and in some applications the best performance is achieved without an additional keypoint detection step [53].

3.4.2 Descriptors

After the 3D keypoint localization, the local geometric information around a keypoint is encoded and stored in a high-dimensional vector (feature descriptor). The main objective is to extract the predominant information of the underlying surface in order to distinguish one local surface region from another. As of today, a lot of different 3D local feature descriptors has been proposed and the choice of the method is not an easy task and usually depends on the characteristics of application. One has to typically choose between feature accuracy, storage requirements and computational efficiency. Based on recent benchmarks, SHOT [128] achieves a good performance in terms of both descriptiveness and computational efficiency [24, 50, 52].

The SHOT descriptor can be considered as a combination of Signatures and Histograms. As a first step the k neighbors of a query point $\vec{p}$ are located and then used to form a weighted covariance matrix $\boldsymbol{C}$:

$$\boldsymbol{C} = \frac{1}{n}\sum_{i=1}^{n}(r - \|\vec{p}_i - \vec{p}\|)\cdot(\vec{p}_i - \vec{p})\cdot(\vec{p}_i - \vec{p})^T, \tag{3.10}$$

where r is the radius of the support region encapsulating the neighboring points $\vec{p}_i$. EVD of the weighted covariance matrix results three orthogonal eigenvectors that define principal axes of the local coordinate system at point $\vec{p}$. The eigenvectors $\vec{v}_1$, $\vec{v}_2$ and $\vec{v}_3$ are sorted based on their eigenvalues and represent the $\vec{x}$-, $\vec{y}$- and $\vec{z}$-axis respectively. To address the sign ambiguity the authors reorient the sign of each eigenvector so that its sign is coherent with the majority of the vectors it is representing. Hence, the $\vec{x}$-axis is oriented as:

$$\begin{aligned} \vec{x} &= \begin{cases} \vec{v}_1, & \text{if } |S_x^+| \leq |S_x^-|. \\ -\vec{v}_1, & \text{otherwise.} \end{cases} \\ S_x^+ &= \{\vec{p}_i : (\vec{p}_i - \vec{p})\cdot\vec{v}_1 \leq 0\} \\ S_x^- &= \{\vec{p}_i : (\vec{p}_i - \vec{p})\cdot\vec{v}_1 > 0\} \end{aligned} \tag{3.11}$$

The same procedure is used to disambiguate the $\vec{z}$-axis. Finally, direction of the $\vec{y}$-axis is calculated through cross-product $\vec{z}\times\vec{x}$. The generated local coordinate system, or the so called *Local Reference Frame (LRF)* [128], is then used to divide the spatial environment of the point $\vec{p}$ with an isotropic spherical grid. For each point in a

local 3D cell an angle is calculated between the points normal and the local $\vec{z}$-axis at the $\vec{p}$ and accumulated to the bin of a local histogram. If the spatial environment is divided into k different cells each having local histograms containing b bins, the resulting final histogram (description) has the length of $k \cdot b$ values.

To improve the discriminativeness of the descriptor, Tombari also proposed a robustified version, called CSHOT [129], that realizes a joint texture-shape 3D feature descriptor. In addition to SHOT, other notable 3D descriptions are for instance Point Pair Feature (PPF) [32], Fast Persistent Histogram Features (FPFH) [114], Unique Shape Context (USC) [128] and Spin Images [66] to name a few.

3.5 Matching

Once the model and scene inputs have been encoded into a feature representation, the next step is to find and match similar features between the two objects, i.e. establish *correspondences*. In particular, we are given two sets of multidimensional feature vectors $\mathscr{F} = \{\vec{f}_i\}_{i=1}^{n}$ and $\mathscr{F}' = \{\vec{f}_i'\}_{i=1}^{m}$ and we find the nearest neighbor feature vector $\vec{f}_1'$ of each query feature vector $\vec{f}$ based on some distance function $d(\cdot)$ defined in the feature space

$$\vec{f}_1' = \arg\min_{\vec{f}' \in \mathscr{F}'} d(\vec{f}, \vec{f}') \tag{3.12}$$

The matches form the initial set of correspondences $\mathscr{C} \subset \mathbb{R}^3 \times \mathbb{R}^3$ where a correspondence $c \in \mathscr{C}$ is parameterized as $c = (\vec{p}, \vec{p}' : \vec{p} \in \mathscr{M}, \vec{p}' \in \mathscr{M}')$. The precision of the matching process is commonly described by the true positive rate or *recall* which is the number of correctly matched point pairs with respect to the total number of correspondences between the two objects. For instance, in Fig. 3.6 there are three different correspondences where only one (c_1) is correctly matched, i.e. giving us recall of 1/3. The choice of the descriptor has high impact on the matching performance as well as the used distance function. In addition, commonly the object model is reconstructed from multiple views and merged into a one dense point cloud. In contrast, the input scene usually contain a single view of the model which means that many of the model features do not have a corresponding point on the input scene, eventually decreasing the precision of the matching process. However, even a relatively simple correspondence filtering procedure (Sec. 3.6) can significantly improve the matching quality.

The search of nearest neighbor (Eq. 3.12) is one of the most demanding steps in the recognition pipeline due to the high dimensionality of the feature vectors. For instance, the SHOT descriptor presented in the previous section has dimensionality of 352. This will create enormous computational complexity and traditional space-partitioning data structures, such as kd-tree, are unpractical [42]. For organizing and searching high dimensional data points new type of techniques have been presented, such as multiple randomized kd-trees [98, 99] that can solve nearest-neighbor queries quickly in an approximated manner. In literature there is not much variation on the choice of the distance function $d(\cdot)$ and the most popular is the Euclidean distance, i.e. $d(\vec{f},\vec{f'}) = \left\| \vec{f} - \vec{f'} \right\|$.

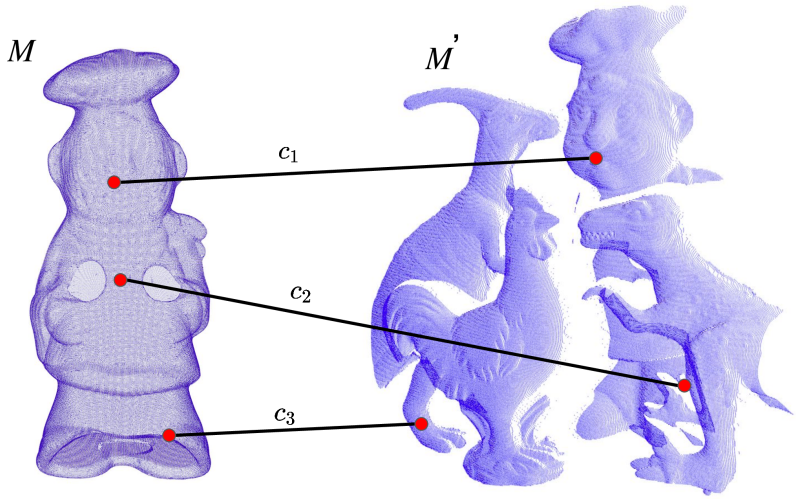


Figure 3.6 Illustration of local feature-based matching, where point-wise matches are established between two geometric shapes. In the figure, $2/3$ of the initial matches are incorrect and the main objective of 3D correspondence filtering is to group the initial matches to inliers (true matches) and outliers (false matches).

3.6 Correspondence filtering

The initial set of correspondences $\mathscr{C}$ contains commonly lots of false matches between the model and scene objects. This is due to number of different reasons such as keypoint localization errors, repetitive surface structure, missing object parts etc.. The present of the false correspondences i.e. *outliers* $\mathscr{C}_{\text{outliers}} \subset \mathscr{C}$ have negative

influence to the pose sampling accuracy and they must be filtered out. In correspondence filtering procedure we are interested of finding a set of *inlier* correspondences $\mathscr{C}_{\text{inliers}} \subset \mathscr{C}$ which represent the correctly matched points between the model and scene. Detailed description and evaluation of different correspondence filtering methods can be found from [144]. In the following, we briefly review the popular methods from the PCL.

3.6.1 Baseline methods

The baseline methods are used in many of the state-of-the-art methods as a preprocessing step due to their almost non-existing overhead. The most straightforward solution is to calculate the similarity score for the closest match and remove the correspondence from the initial set of matches if the Euclidean distance is greater than the threshold value τ_{SC}

$$\left\| \vec{f} - \vec{f}_1' \right\| \geq \tau_{\text{SC}} \ , \tag{3.13}$$

A simple extension to above method is Nearest Neighbor Similarity Ratio (NNSR). It is a popular yet very efficient correspondence grouping method that has been used successfully especially with SIFT feature [85]. The method ranks a correspondence based on the nearest- and second-nearest match

$$\frac{\left\| \vec{f} - \vec{f}_1' \right\|}{\left\| \vec{f} - \vec{f}_2' \right\|} \geq \tau_{\text{NNSR}}, \tag{3.14}$$

where $\tau_{\text{NNSR}} \in [0, 1]$ is the bounded threshold value since $d(\vec{f}, \vec{f}_2') \geq d(\vec{f}, \vec{f}_1')$. The method requires searching of the second-nearest match $\vec{f}_2'$ but this introduces only very limited overhead with a fast search structure [99]. The method has an intuitive interpretation as it filters out matches that cannot be uniquely described i.e. there are two similar feature vectors for the query vector $\vec{f}$.

3.6.2 State-of-the-art

Random Sample Consensus (RANSAC) [38]. RANSAC is a widely used technique for 6D pose estimation [11, 12, 38] adopted from the 2D domain. In contrast to other filtering methods, it can simultaneously estimate the pose of an object and

the set of inliers. The RANSAC algorithm is an iterative process that uses random sampling technique to generate candidate solutions for a model (transformation) that aligns two set of points with a minimum point-wise error. An important parameter of the method is N_{RANSAC} which is the maximum number of pose hypothesis the algorithm samples from the initial correspondence set $\mathscr{C}$. The required number of pose hypothesis for a moderate success rate is usually high as a large fraction of the features are commonly mismatches. When a new iteration step begins, the algorithm selects at least three point pairs from the correspondence set and estimates a transformation matrix aligning the points. Next, all the other matches are aligned using the transformation and if the the Euclidean distance between a transformed point pair is less than τ_{RANSAC}, it is counted as a inlier. The model is considered good if many of the point pairs are counted as inliers. The procedure is repeated until the maximum number of iterations has been reached or the point-wise error is less than the predetermined threshold value.

Spectral Technique (ST) [78]. Leordeanu and Hebert proposed a spectral grouping technique to find coherent clusters from the initial set of feature matches. The method takes into account the relationship between points and correspondences and finally uses an eigen-decomposition to estimate the confidence of a correspondence to be an inlier. First the algorithm creates an affinity matrix $\boldsymbol{M}$ which entries represent weighted links between correspondences. The weights are estimated by calculating the pairwise similarity between two correspondences using a rigidity constraint

$$\boldsymbol{M}(c_i, c_j) = \min\left\{\frac{||\vec{p}_i - \vec{p}_j||}{||\vec{p}'_i - \vec{p}'_j||}, \frac{||\vec{p}'_i - \vec{p}'_j||}{||\vec{p}_i - \vec{p}_j||}\right\} \quad . \tag{3.15}$$

The diagonal elements of the matrix measure the level of individual assignments i.e. how well $\vec{f}_i$ and $\vec{f}'_i$ match. After computing $\boldsymbol{M}$, the principle eigenvector $\vec{v}$ of $\boldsymbol{M}$ is calculated and the location of the maximum value v_k gives the highest confidence of c_k being in the inlier set. Next, all the correspondences conflicting with c_k are removed from the initial set of matches $\mathscr{C}$ and procedure is repeated until $v_k = 0$ or $\mathscr{C}$ is empty and the generated inlier set is returned.

Geometric Consistency Grouping (GC) [21]. The GC approach incrementally builds clusters of correspondences that are geometrically consistent. The grouping

works independently from the feature space and utilizes only the spatial relationship of the corresponding points. The algorithm evaluates the consistency of two correspondences c_i and c_j using a compatibility score

$$\left|\left\|\vec{p}_i - \vec{p}_j\right\| - \left\|\vec{p}'_i - \vec{p}'_j\right\|\right| < \tau_{\mathrm{GC}} \ . \tag{3.16}$$

GC measures an absolute pairwise similarity using the Euclidean distance between the point pairs and assigns correspondences to the same cluster if their geometric inconsistency is smaller than the threshold value τ_{GC}. GC is initialized with a fixed number of clusters each having a seed correspondence. Then for each cluster it iteratively searches correspondences which satisfy the compatibility score (Eq. 3.16), mark them as visited and continue the process until all the correspondences are evaluated. Finally, all the cluster sets can be optionally refined using RANSAC. In principle, the GC algorithm can return more than one cluster. For pose estimation the cluster with the largest number of correspondences is a good choice [**P2**].

Hough Grouping (HG) [127]. The key idea of the Hough 3D correspondence grouping is to iteratively cast votes for object location in the Hough parameter space. Each correspondence accumulates a bin in the Hough space and at the end of the process the bins with the biggest number of casts represent the most likely pose candidate and the correspondences contributed to the bin are accepted. Each bin represents a single pose instance candidate and therefore all the correct correspondences vote to a same bin which gets quickly accumulated. To make correspondence points invariant to rotation and translation between the model and scene, every point is associated with LRF. The most important parameters of the method are the Hough accumulator bin size and the minimum number of votes needed to infer the presence of a model instance.

Search of Inliers (SI) [13]. The SI method is based on two consecutive processing stages, *local voting* and *global voting*. At the end, the votes are accumulated to form a quantitative indicator (number of votes) to denote the degree of trust for each correspondence. The first voting step performs local voting, where locally selected correspondence pairs are selected from the model and scene, and the score is computed using their pair-wise similarity score $s_{local}(\vec{p})$ for each 3D point $\vec{p}$. The second global voting stage samples point correspondences, estimates a transforma-

tion and gives a global score to the points correctly aligned outside the estimation point set: $s_G(\vec{p})$. The final score $s(\vec{p})$ is computed by integrating both local and global scores, and an adaptive threshold between inliers and outliers is automatically found by Otsu's bimodal distribution thresholding. The inlier set is used for final pose estimation.

3.7 Estimating the pose from correspondences

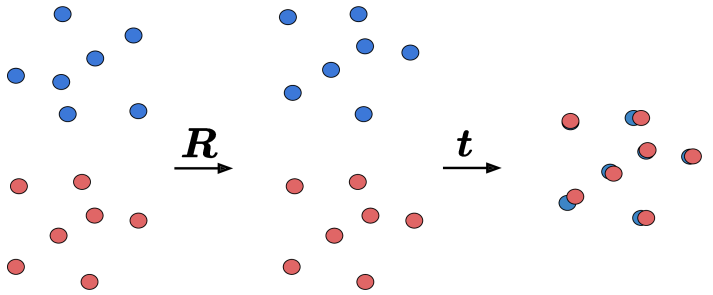


Figure 3.7 Transformation matrix $\Upsilon = [R|\vec{t}]$ rotates and translates the target points (blue dots) to align with query points (red dots).

As stated in the introduction the main target in pose estimation is to find the transformation matrix Υ that best describes the target object position and orientation in the input image (see Eq. 3.1). This is formally an optimization problem and various different solution has been proposed. In correspondence-based pose estimation, we can simplify the problem to

$$\Upsilon = \arg\min_{\hat{\Upsilon}} \sum_{\vec{p},\vec{p}' \in \mathscr{C}_{\text{inliers}}} ||\hat{\Upsilon}(\vec{p}) - \vec{p}'|| \ . \tag{3.17}$$

where the main objective is to find the transformation between two sets of corresponding 3D points that minimizes the Euclidean distance between the paired points (Fig. 3.7). This corresponds to a linear least-squares problem that can be solved robustly using SVD [67].

In a typical pose estimation pipeline the correspondences provide only an initial pose that roughly registers the two sets of points. The initial pose is commonly refined using the ICP algorithm [23], which iteratively refines the initial pose by

repeatedly generating pairs of points on the point cloud and minimizing an error metric. Instead of using only the feature point matches, the algorithm utilized all the model points. Because the ICP calculates the distance between the neighboring points, the algorithm requires a relatively good starting point in advance or the error will get stuck in non-optimal local minimum giving an invalid pose. The input for the algorithm is the initial guess $\boldsymbol{\Upsilon}_{\text{initial}}$ of the transformation matrix aligning the two surfaces and the maximum iteration count N_{ICP}.

3.8 Pose Estimation Metric for Robotic Manipulation

In the previous sections we have described all the necessary steps that are needed to estimate the pose of a rigid object. The next step is to validate the estimated pose i.e. measure its quantitative performance on a specific task. Several evaluation metrics have been proposed in literature for measuring the fitness of the estimated 6D object pose and the choice is not always trivial. For instance, in augmented reality one of the most important factors is the witnessed visual experience. In this case, it is enough that only the part of an object that is visible for the user is correctly estimated for virtual content alignment. In contrast, the fundamental task in robotic manipulation is to interact with objects in the environment and succeeding in such a task requires accurate positioning of the robot end effector with respect to the object, especially when interacting with objects having complex shape.

3.8.1 Background

The ADC metric proposed by Hinterstoisser [56] is one of the most widely used 6D pose error metric. The metric calculates the distance between object model points transformed by the ground truth pose $\hat{\boldsymbol{\Upsilon}}$ and estimated pose $\boldsymbol{\Upsilon}$. The distance score is defined as

$$\epsilon_{ADC} = \frac{1}{|\mathcal{M}|} \sum_{\vec{p} \in \mathcal{M}} ||\hat{\boldsymbol{\Upsilon}}(\vec{p}) - \boldsymbol{\Upsilon}(\vec{p})|| \ . \tag{3.18}$$

The metric is easy to interpret as it directly measures the fitness of the surface alignment of the transformed objects.

In [14, 32] the performance of a pose estimation algorithm is rated using a model-

independent error function, called *translation and rotational error* (TRE)

$$\epsilon_T = \left\| \boldsymbol{t} - \hat{\vec{t}} \right\|, \quad \epsilon_R = \arccos\left(\frac{Tr\left(\hat{\boldsymbol{R}}\boldsymbol{R}^{-1}\right) - 1}{2} \right) \tag{3.19}$$

The rotational error ϵ_R is computed as the geodesic distance between the estimated and predicted rotation and gives the minimal angle needed to rotate $\hat{\boldsymbol{R}}$ into $\boldsymbol{R}$. The Euclidean distance is used to calculate the translation error ϵ_T. Compared to other methods the error is fast to compute as it directly compares the two poses without utilizing thousands of points of the model point cloud.

However, both of the metrics do not consider the pose ambiguity problem which refers to a situation where an object instance in the input image can be perfectly described by several different poses. The object pose can be ambiguous from different viewpoints due to shape symmetries, occlusion and repetitive textures. This is a common problem as typical RGB-D datasets proposed in literature include traditional house hold objects that exhibit shape ambiguities and repetitive patterns causing their visual appearance to be very similar. However, the datasets do not take into account these ambiguities and provide only one unique ground truth pose annotation for each test scene. This is problematic as visually correct pose can get high prediction error and comparison of different pose estimation methods becomes difficult. As it is not practical to generate hundreds of ground truth poses to cover all the similar poses, the research community has focused on generating efficient evaluation metrics that assign same penalty to geometrically indistinguishable poses.

To address the ambiguity problem, Hinterstoisser [56] proposed an extended similarity metric, where instead of calculating the distance between corresponding points the distance was measured between the nearest pair of points (ADN)

$$\epsilon_{ADN} = \frac{1}{|\mathscr{M}|} \sum_{\vec{p}_1 \in \mathscr{M}} \arg\min_{\vec{p}_2 \in \mathscr{M}} ||\hat{\boldsymbol{\Upsilon}}(\vec{p}_1) - \boldsymbol{\Upsilon}(\vec{p}_2)|| \ . \tag{3.20}$$

In addition to the pose ambiguities caused due to object shape and appearance, self-occlusion and occlusion by another objects can induce ambiguities in visual appearance, as shown in Fig. 3.8. This is not considered in the ADN metric and for instance in the figure the error is calculated over the mug handle although it is not visible from the camera point of view-point. To address this issue, Hodan et. al [59, 60] have recently proposed a new evaluation metric for 6D pose estimation, called

Visual Surface Discrepancy (VSD). VSD calculates the pixel-wise error only over the visible part of the model surface and it is defined as

$$e_{VSD} = \frac{1}{|\hat{\mathcal{V}} \cup \mathcal{V}|} \sum_{\vec{p} \in \hat{\mathcal{V}} \cup \mathcal{V}} \begin{cases} 0 & \text{if } \vec{p} \in \hat{\mathcal{V}} \cap \mathcal{V} \wedge |\hat{\mathcal{D}}(\vec{p}) - \mathcal{D}(\vec{p})| < \tau_{VSD} \\ 1 & \text{otherwise} \end{cases}, \quad (3.21)$$

where $\mathcal{D}$ and $\hat{\mathcal{D}}$ are distance maps of the object model transformed by the ground truth and estimated pose, respectively. Threshold τ_{VSD} is used to penalize the misalignment between the distance maps. In addition, the method requires the visibility masks $\hat{\mathcal{V}}$ and $\mathcal{V}$ in order to compute the set of pixels which are visible from the model $\mathcal{M}$.

Figure 3.8 Example causes of pose ambiguity. Different poses of the mug cannot be recognized due to self-occlusion (left). Pose ambiguity introduced due to occlusion by another object (right).

The evaluation metrics mentioned above ranks the estimated pose solely based on the difference of two pose matrices. However, the success on a robotic task depends on many factors beyond the accuracy of estimating the 6D pose parameters: i) the manipulated object and its properties (material, weight, dimensions), ii) the selected gripper (including fingers), iii) the selected grasp point and grasping maneuver, and iv) the task itself (bin picking vs. precision wrenching). There have been attempts to report success rates for a particular setup [49, 105, 119, 137], but these require implementation of the same physical setup for method evaluation. The robot-vision community would benefit from datasets and evaluation metrics that can measure the actual performance of the estimated pose in a specific task without requiring a physical setup.

In the following, we propose such a metric. The presented method is described in publication [**P6**], proposing a completely new benchmark for object pose estimation. The metric is based on a statistical formulation of a successfully conducted robotic task ($X = 1$) given the estimated object pose $\hat{\boldsymbol{\Upsilon}}$. Concretely, the estimated object pose is converted and parametrized as the 6D pose $\hat{\vec{\theta}}$ of the robot gripper in the object-relative coordinate space and evaluated using a conditional probability metric $P(X=1|\hat{\vec{\theta}})$. Interpretation of the metric is intuitive: 0.9 means that on average ninety out of one hundred attempts succeed with the given pose estimate. The probabilistic metric is trained using a number of real grasp samples $\vec{\theta}_i$ which are generated using a real physical robot and an automatic sampling procedure (short video example [5]).

3.8.2 Probability of completing a programmed task

The success of a robot to complete its task is a binary random variable $X \in \{0,1\}$ where $X=1$ denotes a successful attempt and $X=0$ denotes an unsuccessful attempt (failure). Therefore, X follows the Bernoulli distribution, $P(X|p) = p^X(1-p)^{1-X}$, with complementary probability of success and failure: $E(X) = P(X=1) = 1 - P(X=0)$, where E denotes the mathematical expectation. The pose is defined by 6D pose coordinates $\vec{\theta} = (t_x, t_y, t_z, r_x, r_y, r_z)^T$ where the origin is the object centric coordinate frame. The translation vector $(t_x, t_y, t_z)^T \in \mathbb{R}^3$ and 3D rotation $(r_x, r_y, r_z)^T \in SO(3)$ both have three degrees of freedom. The rotation is in axis-angle representation, where the length of the 3D rotation vector is the amount of rotations in radians, and the vector itself gives the axis about which to rotate. Adding pose to the formulation makes the success probability a conditional distribution and expectation a conditional expectation. The conditional probability of a successful attempt is

$$p\left(\vec{\theta}\right) = E(X|\vec{\theta}) = P(X=1|\vec{\theta}) = 1 - P(X=0|\vec{\theta}) \ . \tag{3.22}$$

The maximum likelihood estimate of the Bernoulli parameter $p \in [0,1]$ from N homogeneous samples y_i, $i=1,\ldots,N$, is the sample average

$$\hat{p}_{\mathrm{ML}} = \frac{1}{N}\sum_{i=1}^{N} y_i \ , \tag{3.23}$$

[5] https://youtu.be/g4e_-p4fTEI

where homogeneity means that all samples are realization of a common Bernoulli random variable with unique underlying parameter p. However, guaranteeing homogeneity would require that the samples $\{y_i, i=1,\dots,N\}$ were either all collected at the same pose $\vec{\theta}_1=\cdots=\vec{\theta}_N$, or for different poses that nonetheless yield the same probability $p(\vec{\theta}_1)=\cdots=p(\vec{\theta}_N)$, i.e. it would require us either to collect multiple samples for each $\vec{\theta}\in SE(3)$ or to know beforehand p over $SE(3)$ (which is what we are trying to estimate). This means that in practice p must be estimated from non-homogeneous samples, i.e. from $\{y_i, i=1,\dots,N\}$ sampled at pose $\left\{\vec{\theta}_i, i=1,\dots,N\right\}$ which can be different and having different underlying $\left\{p(\vec{\theta}_i), i=1,\dots,N\right\}$.

The actual form of p over $SE(3)$ is unknown and depends on many factors, e.g., the shape of an object, properties of a gripper and a task to be completed. Therefore it is not meaningful to assume any parametric shape such as the Gaussian or uniform distribution. Instead, we adopt the Nadaraya-Watson non-parametric estimator which gives the *probability of a successful attempt* as

$$\hat{p}_{\vec{h}}\left(\vec{\theta}\right)=\frac{\sum_{i=1}^{N} y_i K_{\vec{h}}\left(\vec{\theta}_i-\vec{\theta}\right)}{\sum_{i=1}^{N} K_{\vec{h}}\left(\vec{\theta}_i-\vec{\theta}\right)}\ , \tag{3.24}$$

where $\vec{\theta}_i$ denotes the poses at which y_i has been sampled and $K_{\vec{h}}:\mathscr{E}\to\mathbb{R}^+$ is a non-negative multivariate kernel with vector scale $\vec{h}=\left(h_{t_x},h_{t_y},h_{t_z},h_{r_x},h_{r_y},h_{r_z}\right)^T>0$.

In this work, $K_{\vec{h}}$ is the multivariate Gaussian kernel

$$\begin{aligned}K_{\vec{h}}\left(\vec{\theta}\right)=G\left(\tfrac{t_x}{h_{tx}}\right)G\left(\tfrac{t_y}{h_{ty}}\right)G\left(\tfrac{t_z}{h_{tz}}\right)\sum_{j\in\mathbb{Z}}G\left(\tfrac{r_x+2j\pi}{h_{rx}}\right)\cdot\\ \sum_{j\in\mathbb{Z}}G\left(\tfrac{r_y+2j\pi}{h_{ry}}\right)\sum_{j\in\mathbb{Z}}G\left(\tfrac{r_z+2j\pi}{h_{rz}}\right),\end{aligned} \tag{3.25}$$

where G is the standard Gaussian bell, $G(\theta)=(2\pi)^{-\frac{1}{2}}e^{\frac{1}{2}\theta^2}$. The three sum terms in (3.25) realize the modulo-2π periodicity of $SO(3)$.

The performance of the estimator (3.24) is heavily affected by the choice of $\vec{h}$, which determines the influence of samples y_i in computing $\hat{p}_{\vec{h}}\left(\vec{\theta}\right)$ based on the difference between the estimated and sampled poses $\vec{\theta}$ and $\vec{\theta}_i$. Indeed, the parameter $\vec{h}$ can be interpreted as reciprocal to the bandwidth of the estimator: too large $\vec{h}$ results in excessive smoothing whereas too small results in localized spikes.

To find an optimal $\vec{h}$, we use the leave-one-out (LOO) cross-validation method. Specifically, we construct the estimator on the basis of N–1 training examples leaving out the i-th sample:

$$\hat{p}^{\text{LOO}}_{\vec{h}}\left(\vec{\theta}, i\right) = \frac{\sum_{j \neq i} y_j K_{\vec{h}}\left(\vec{\theta}_j - \vec{\theta}\right)}{\sum_{j \neq i} K_{\vec{h}}\left(\vec{\theta}_j - \vec{\theta}\right)} \ .$$

The likelihood of y_i given $\hat{p}^{\text{LOO}}_{\vec{h}}\left(\vec{\theta}_i, i\right)$ is either $\hat{p}^{\text{LOO}}_{\vec{h}}\left(\vec{\theta}_i, i\right)$ if $y_i = 1$, or $1 - \hat{p}^{\text{LOO}}_{\vec{h}}\left(\vec{\theta}_i, i\right)$ if $y_i = 0$. We then select $\vec{h}$ that maximizes the total LOO log-likelihood over the whole set S_y:

$$\hat{\vec{h}} = \arg\max_{\vec{h}} \sum_{i|y_i=1} \log\left(\hat{p}^{\text{LOO}}_{\vec{h}}\left(\vec{\theta}_i, i\right)\right) + \sum_{i|y_i=0} \log\left(1 - \hat{p}^{\text{LOO}}_{\vec{h}}\left(\vec{\theta}_i, i\right)\right).$$

3.8.3 Sampling the pose space

Section 3.8.2 provides us a formulation of the probability of successful robotic manipulation given the object relative grasp pose $P(X = 1|\vec{\theta})$. The practical realization of the probability values is based on Nadaraya-Watson non-parametric kernel estimator that requires a number of samples in various poses $\vec{\theta}_i$ and information of success $y_i = 1$ or failure $y_i = 0$ for each attempt. In this stage, a physical setup is needed for sampling, but the users of the benchmark do not need to replicate the setup - they need only the pre-computed probability densities provided with the benchmark. For practical reasons we make the following assumptions:

- We define a canonical grasp pose with respect to a manipulated objects which is select based on the object intrinsic parameters (i.e. the distribution of mass) and task requirements (i.e. on which way the object is being installed). During the sampling procedure the canonical pose is located using a 2D marker.
- We sample the pose space around the canonical grasp pose, and therefore $\vec{\theta} = (t_x, t_y, t_z, r_x, r_y, r_z)^T$ defines $SE(3)$ "displacement" from the canonical grasp pose. Sampling was started by first finding the sampling limits of each dimension and then sampling within the limits. The limits were found by manually guiding the end effector away from the canonical grasp pose along each dimension until the task execution always failed.

With the help of these assumptions we are able to define a sampling procedure that can record samples and their success or failures automatically. The main limitation of this approach is that the pose space is sampled only near the canonical grasp pose which is not guaranteed to be the best option in every scenario. For instance, the grasp pose might be unreachable due to robot's kinematic constraints or obstructing objects. In our work, we assume that the canonical grasp pose is always reachable. In addition, the underlying probability distribution for successful attempts depends heavily on the robot setup, robotic task and grasping strategy. For instance, changing the robot gripper would require a new set of training samples to be collected in order to create a valid performance metric for that specific setup. On the other hand, all the code will be made publicly available to promote construction of novel benchmarks with different physical setups in other laboratories.

Coordinate transformations. In the work, a coordinate transformation $\boldsymbol{T}_B^A$ denotes a 4×4 homogeneous transformation matrix that describes the position of the frame B origin and the orientation of its axes, relative to the reference frame A.

For a practical implementation used in our experiments the transformation components are (Figure 3.9):

- $\boldsymbol{T}_{grasp}^{marker}$ – a constant transformation from the canonical grasp pose to the marker frame;
- $\boldsymbol{T}_{marker}^{sensor}$ – computed transformation from the marker frame to the sensor frame;
- $\boldsymbol{T}_{sensor}^{effector}$ – a constant transformation from the sensor frame to the robot end effector frame (camera is attached to the end effector);
- $\boldsymbol{T}_{effector}^{world}$ – computed transformation from the end effector frame to the world frame (robot origin).

The world frame is fixed to the robot frame (i.e. center of the robot base) and programming is based on the tool point that is the end effector frame. The coordinate transformation $\boldsymbol{T}_{effector}^{world}$ can be automatically calculated using the joint angles and known kinematic equations. $\boldsymbol{T}_{sensor}^{effector}$ is computed using the standard procedure for hand-eye calibration with a printed chessboard pattern [103]. Automatic and accurate estimation of the object pose during the sampling is realized by attaching an artificial 2D markers to the manipulated objects (see Fig. 3.10 for an example). For a calibrated camera the ArUco library [44] provides an accurate real-time pose of the

marker with respect to the sensor frame T^{sensor}_{marker}. The constant offset T^{marker}_{grasp} from the marker to the actual grasp pose is object-marker specific and it is estimated manually by hand-guiding the end effector to the desired grasp location on the object (canonical grasp pose) and then measuring the difference between this pose and the marker pose:

$$T^{marker}_{grasp} = \left(T^{world}_{marker}\right)^{-1} T^{world}_{grasp}.$$

During the sampling procedure, the canonical grasp pose is then calculated with respect to world frame as:

$$T^{world}_{grasp} = T^{world}_{effector} T^{effector}_{sensor} T^{sensor}_{marker} T^{marker}_{grasp} \tag{3.26}$$

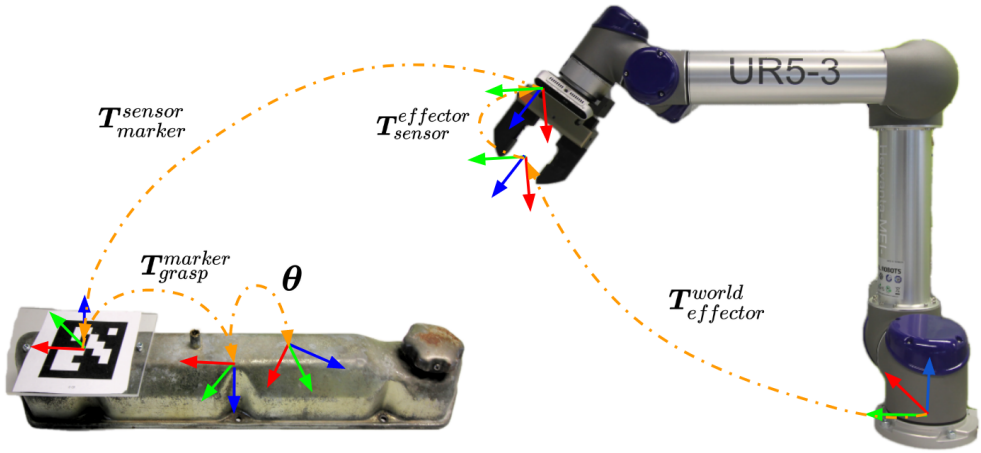


Figure 3.9 Coordinate frames used in random sampling of poses for assembly tasks.

Finally, samples around the canonical grasp pose are generated from

$$\hat{T}^{world}_{grasp} = T^{world}_{grasp} \Phi(\vec{\theta}) \tag{3.27}$$

where the operator $\Phi(\cdot)$ converts the 6D pose vector to a 4×4 matrix representation

$$\Phi(\vec{\theta}) = \begin{bmatrix} R_{3\times3} & \vec{t} \\ \vec{0} & 1 \end{bmatrix}. \tag{3.28}$$

The generated pose sample is defined in the vicinity of the canonical pose by the translation shift $\vec{t} = (t_x, t_y, t_z)^T$ and rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ constructed from the axis-angle vector $(r_x, r_y, r_z)^T$.

Detection of failures. Each of the manipulated objects has a predefined position and orientation how it should be installed, i.e. *ground truth installation pose*, with respect to the target object. For instance most of the motor parts have to be placed on the motor block precisely in order to fasten the parts with screws. The robot task is to bring the part to this pose and finally release the part by opening the gripper fingers. In addition, using excessive force during the task can cause damage to the manipulated objects. In the work there are two sources of information for detecting manipulation failures: 1) too large difference between the installation pose of the manipulated object and the corresponding ground truth and 2) too large wrench torque at the end effector at any moment of task execution (e.g. due to collisions), including grasping, carrying and installation. Thresholds for the above are task specific and in our experiments they were manually set based on preliminary experiments.

For evaluation of the success of the part installation in the terms of correct pose, the translation and rotational error metric was adopted (see Section 3.8.1). The error is calculated using the installation pose $\hat{\boldsymbol{\Gamma}} = \left[\hat{\boldsymbol{R}} \mid \hat{\vec{t}}\right]$ measured using the marker attached to the manipulated work part and the ground truth installation pose $\boldsymbol{\Gamma} = \left[\boldsymbol{R} \mid \vec{t}\right]$. The installation is successful if the difference is less than τ_t (translation) and τ_r (rotation).

The torque is used to detect if the robot collides with its environment during the task execution. In addition, if the robot places the object to the correct position with too high wrench the whole task is considered as an unsuccessful attempt. The external wrench is computed based on the error between the joint torques required to stay on the programmed trajectory and the expected joint torques. The robot's internal sensors provide the torque measurements $\boldsymbol{F} = (f_x, f_y, f_z)$, where f_x, f_y and f_z are the forces in the axes of the robot frame coordinates and measured in Newtons. For each task the limit f_{max} was manually set for each operation stage using preliminary experiments and violating the threshold, i.e $||\boldsymbol{F}|| > f_{max}$, was recorded as failure.

Tasks. The experiments were conducted on practical tasks selected from the production line of a local engine manufacturing company. The selected tasks were:

(Task 1) installation of a motor cap 1, (Task 2) installation of a motor frame and (Task 3) installation of a motor cap 2 (different engine model). The fourth task (Task 4) is different from others: picking and dropping a part to a container (the *faceplate* part from the Cranfield assembly benchmark). As Task 4 does not require precise manipulation, the task requires less accurate pose than the others. Cranfield faceplate was selected since its 3D model is publicly available and the part is used in robot manipulation studies. The tasks were programmed by an experienced engineer who also carefully selected the grippers and fingers. The engineer was instructed that accurate pose is always available.

Setup. In Fig. 3.10 is illustrated the robotic setup used in our experiments. The setup consisted of a model 5 Universal Robot Arm (UR5) and a Schunk PGN-100 gripper. The gripper operates pneumatically and was configured to have a high gripping force (approximately 600N) to prevent object slippage. In addition, the gripper had custom 3D printed fingers plated with rubber. For visual perception, an Intel RealSense D415 RGB-D sensor was secured on a 3D printed flange mounted between the gripper and the robot end effector. All the in-house made 3D prints were made using nylon reinforced with carbon fiber to tolerate external forces during the experiments. The computation was performed on a single laptop with Ubuntu 18.04. All

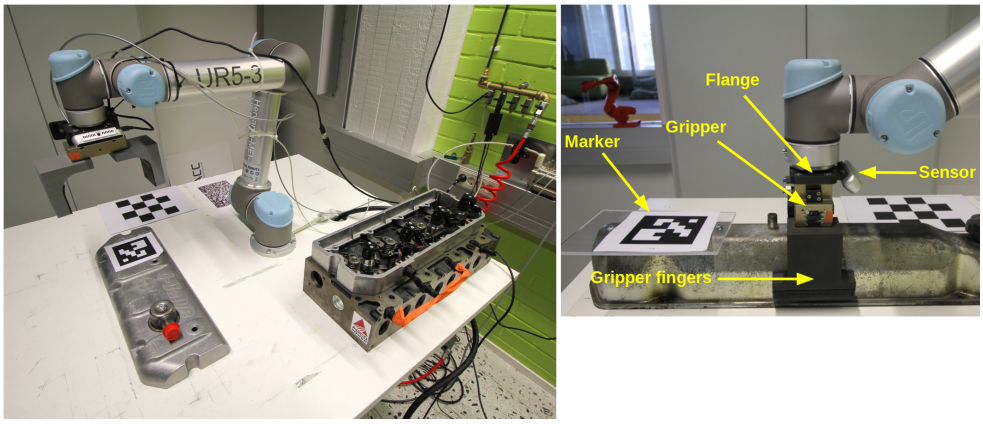


Figure 3.10 The experimental robot setup to sample the pose space of the engine cap 1. The task is to grasp and accurate assembly the cap to the engine mainframe.

tasks and the canonical grasp poses were validated by executing the task 100 times with pose obtained using the 2D patterns (Section 3.8.3). No failures occurred dur-

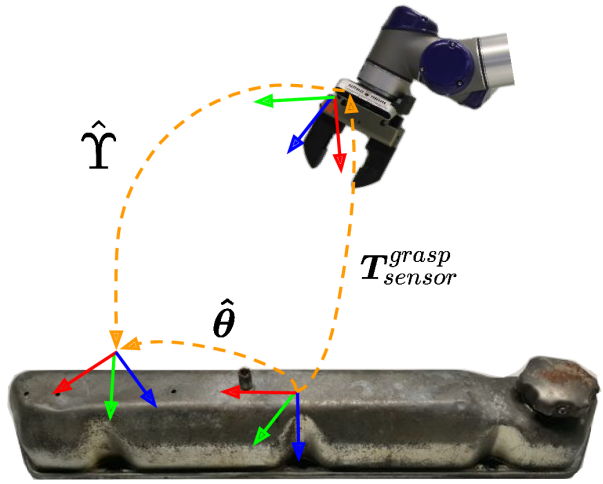


Figure 3.11 Coordinate frames in the evaluation procedure.

ing the validation. On average, successful executions took 45-55 seconds and in 24 hours the robot was able to execute approximately 1,100 attempts. The setup was able to automatically recover from most of the failure cases (dropping the object, object collision, etc.), however, if the marker was occluded by the environment or if the manipulated object got jammed against internal parts of the motor, the system was restarted by a human operator.

3.8.4 Performance indicator

The main goal of the proposed performance metric is to calculate task success probability for the estimated object pose. The probabilities were computed around the canonical grasp pose of each object and therefore the estimated object pose has to be transformed to the same pose space. The corresponding object-relative grasp pose of the pose estimate $\hat{\Upsilon}$ is calculated as

$$\vec{\hat{\theta}} = \mathbf{\Phi}^{-1}\left(T^{grasp}_{sensor}\hat{\Upsilon}\right) , \tag{3.29}$$

where the transformation matrix T^{grasp}_{sensor} defines the canonical grasp pose with respect to the sensor coordinate system as shown in Fig. 3.11. The $\mathbf{\Phi}^{-1}(\cdot)$ operator converts the 4×4 pose matrix to 6D vector representation. Finally, the task success is evaluated using the proposed metric as $P(X=1|\vec{\hat{\theta}})$.

3.8.5 Model validation

The probability model $P(X = 1|\vec{\theta})$ in Section 3.8.2 was fitted using the sampling procedure in Section 3.8.3. For all tasks approximately 3,300 valid samples were generated around task canonical poses. The estimated probability models were validated by sampling each dimension of $\vec{\theta}$ separately on grid points and executing the task ten times on each point with real robot. The averaged task success rate on real robot was then compared against the proposed models and the estimated probabilities matched well as can be seen in Fig. 3.12.

Next, the proposed metric was compared against the ADC metric in controlled experiments. For each task we generated a synthetic set $\{\hat{\boldsymbol{\Upsilon}}_i\}_{i=1}^{N}$ of 6D poses where each pose $\hat{\boldsymbol{\Upsilon}}_i$ differs from the ground truth pose $\boldsymbol{\Upsilon} = \boldsymbol{I}$ either by rotation or translation along a single axis. The success probability was estimated using the procedure in Sec. 3.8.4 and the ADC score as described in Sec. 3.8.1. The results are shown in Fig. 3.13. It is important to notice that the ADC error cannot take into account the intrinsic parameters of the tasks and assigns the the same performance score for the hardest (Task 1) and easiest task (Task 4) over the same set of object poses. In contrast, we can see that our proposed performance metric can clearly indicate that Task 1 has significantly less tolerance in the pose estimation errors and thus is much harder to conduct. Moreover, the proposed metric can measure the turning point after which the success probability drops quickly from 1.0 to 0.0 where the as ADC error regrades linearly even after these points and is thus uninformative. The difference between the two metrics is further illustrated in the success probability vs. ADC scatter plots of all four tasks in Fig. 3.14.

3.9 Summary

In this chapter, the 6D pose estimation pipeline was described along with the contributions. In specifically, we focused on point cloud and correspondence-based approaches, where point pair matches between two input objects are predicted based on local support, i.e. based on geometric surface properties of a small region. Finally, a pose (transformation matrix) is estimated that aligns the corresponding point pairs based on some distance function. Usually the pose search is guided by exploiting geometric constrains between the correspondences and using robust techniques such

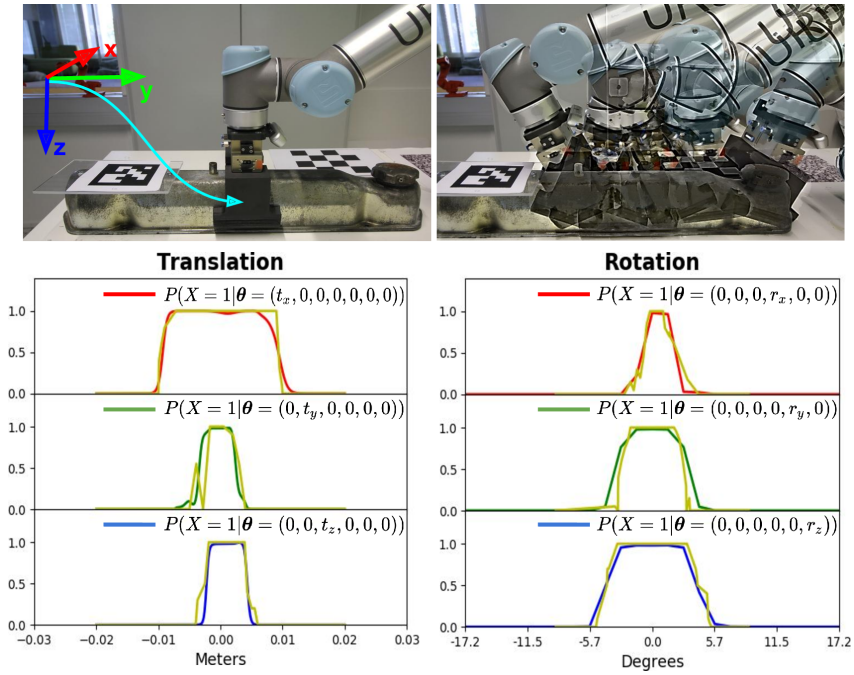


Figure 3.12 Engine cap used in our experiments. The coordinate system is object centric (top left) and pose samples are taken around a canonical grasp pose. Below are the estimated (the red, green and blue lines) and validated success probabilities (yellow line) on the six main axes (three translations and three rotations) in vicinity of the canonical grasp pose.

as the well known RANSAC or point clustering.

During the chapter, we descried two alternative methods to improve the performance of a pose estimation pipeline: *curvature filtering* and *region pruning*. In our preliminary experiments, the robustifying methods were able to identify a robust sub-set of points against estimation failures and consistently improved the three correspondence based methods: GC [21], HG [127] and SI [13]. However, there was no clear winner between the two robustifying methods and a new study was conducted with larger dataset. The main result on the new dataset was that the GC with optimized parameters outperformed all the other methods on most of the object classes. However, unlike in our previous work with limited data, the two robustifying approaches did not provide systematic improvement after the meta parameters of the pose estimation methods were tuned.

In the final section of the chapter, we discussed different techniques for measuring the estimated object pose and introduced completely new metric for robotic manip-

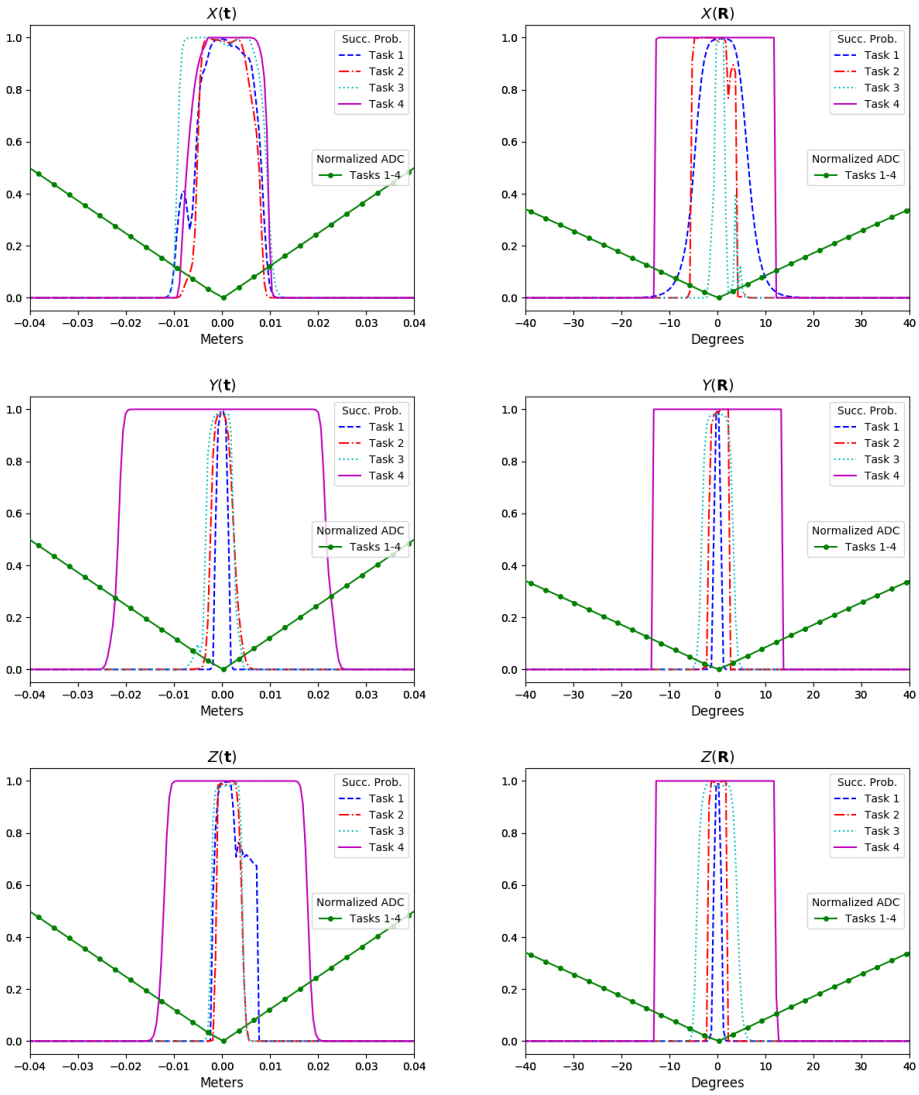


Figure 3.13 ADC and success probability from controlled experiments for all the tasks (Task 1 – Task 4). Effect of rotation (left column) and translation (right column) to the ADC and success probability.

ulation. In our experiments we demonstrated how the popular error measure, ADC, poorly indicates success in robot manipulation tasks and is therefore uninformative. As a novel solution, we proposed a probabilistic metric that measures the true success rate without the physical setup and provides basis for more realistic evaluation

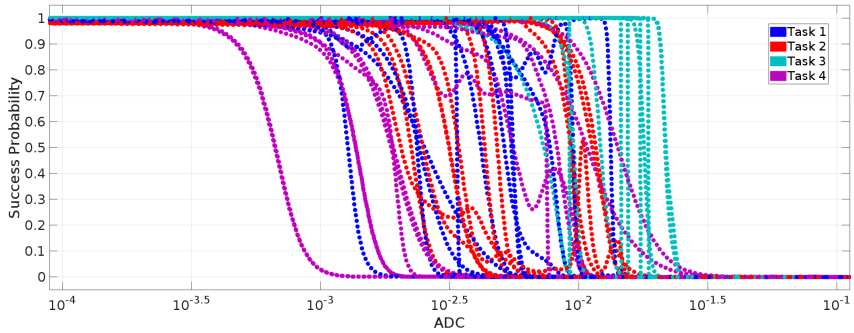


Figure 3.14 Success Probability vs. ADC scatterplot from controlled experiments for all the tasks (Task 1 – Task 4). The scatterplot shows that the ADC does not reflect the success probability, except for extreme and trivial cases of failure or success; the two measures cannot be put in correspondence to each other not even through a nonlinear mapping.

of object pose estimation methods.

4 SAFE HRC IN INDUSTRIAL MANUFACTURING

4.1 Introduction

For decades, industrial robots have been irreplaceable resource for manufacturers, being efficient in repeatable and simple tasks. In isolated workcells the robots can operate without any external sensors and apply simple strategies to succeed in tasks, such as in welding and part feeding. However, demand for more flexible and collaborative systems is rising and currently the industrial manufacturing is going towards a new industrial revolution, the so-called *Industry 4.0*. Human-robot collaboration (HRC) will have an important role in the shift and this evolution means breaking with the established safety procedures as the separation of workspaces between robot and human operator is removed. However, this will require special care for human safety as the existing industrial standards and practices are based on the principle that operator and robot workspaces are separated and violations between them are monitored.

HRC has been active in the past to realize the future manufacturing expectations and made possible by several research results obtained during the past five to ten years within the robotics and automation scientific communities [140]. In particular, this has involved novel mechanical designs of lightweight manipulators, such as the Universal Robot family[1] and KUKA LBR IIWA[2]. Due to the lightweight structure, slow speed, internal safety functions and impact detection, the robots are considered a more safe solution for close proximity work than traditional industrial robots. The collaborative robots can be inherently safe, but the robotic task can create safety hazards for instance by including sharp or heavy objects that are carried at high speed.

[1] https://www.universal-robots.com/

[2] https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa

In order to guarantee the safety of the human co-worker, a large variety of external multi-modal sensors (camera, laser, structured light etc.) have been introduced and used in robotics applications to prevent collisions [51, 110]. Moreover, in order to transfer research solutions from the research lab to real industrial environments they need to comply with strict safety standards.

4.1.1 HRC in manufacturing

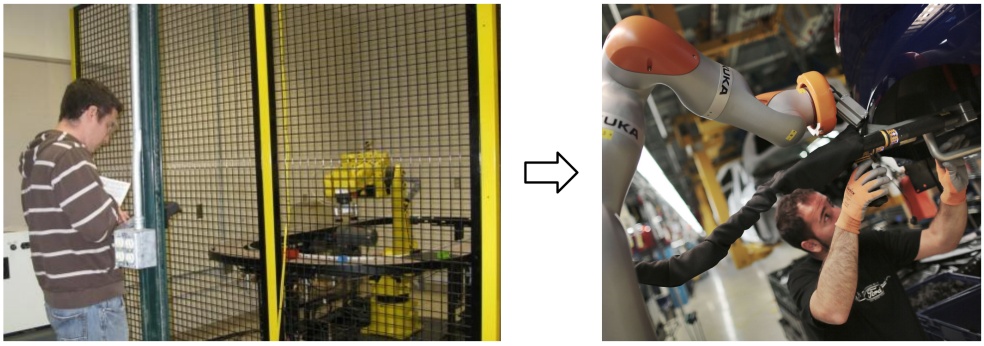


Figure 4.1 Manufacturing is moving from isolated robotic cells towards HRC where humans can work side-by-side with robots in close proximity.

HRC in the manufacturing aims at creating work environments where humans can work side-by-side with robots in close proximity (see Fig. 4.1). In such setup, the main goal is to achieve efficient and high quality manufacturing processes by combining the best of both worlds: strength, endurance, repeatability and accuracy of robots complemented by the intuition, flexibility and versatile problem solving skills of humans. During a collaborative task, the first priority is to ensure safety of the human co-worker. Vision sensors have been an efficient and popular choice to gain information from the surrounding environment, which is crucial for safe trajectory planning and collision avoidance. Other sensing modalities, such as pressure/force, can be combined with visual information to enhance the local safety sensing [89]. In addition to the safety aspect, one of the key challenges in industrial HRC is the interaction and communication between the human and robot resources [130]. According to Liu and Wang [83] the ICT system should be able to provide information feedback and support for the human co-worker during a collaborative task. In industrial settings, the physical environment (i.e. floor, tables) can be used as a medium

where task-related information, such as boundary of safe operation space or user interface components can be projected.

In the literature, several recent works have demonstrated their HRC systems on real industrial manufacturing tasks, where both aspects, safety and communication, are considered. Vogel et al. [139] presented a collaborative screwing application where a projector-camera based system was used to prevent collision and display interaction and safety-related information during the task. In [12] the authors proposed a wearable AR-based interface integrated to an off-the-shelf safety system. The wearable AR supports the operator on the assembly line by providing virtual instructions on how to execute the current task in the form of textual information or 3D model representation of the parts. The integrated interface in [46] was utilized in an automotive assembly task where a wheel group was installed as a shared task. De Gea Fernández [45] and Magrini [86] fused sensor data from different sources (IMU, RGB-D and laser) and a standardized control and communication architecture was used for safe robot control. Human actions and intentions were recognized through hand gestures and the systems were validated in a real industrial task from the automotive industry.

While the mentioned implementations are good examples of safe HRC in manufacturing, the works are mainly technological demonstrations and do not provide data from qualitative or quantitative evaluations that could further emphasize the need of HRC. Similar to our works [**P3**, **P4**, **P5**], an AR-based approach was utilized in car door assembly and evaluated against two baseline methods, imitating the current practices from industry [43]. From the experiments, quantitative (efficiency and effectiveness of the task completion) as well as qualitative data (human-robot fluency, trust in robot etc.) were measured through recordings and questionnaires, respectively.

4.1.2 Collaborative robots

The collaborative robots, or *cobots*, are specifically designed for direct interaction and communication with a human. The collaboration is commonly done in close proximity where the human and robot are doing separate or common task side-by-side. Collaborative robots usually combine all or some of the following characteristics:

- The robots are designed to be safe around human co-workers by force limiting

or internal sensors that prevent collision or minimize injury during collaboration.

- Much more lightweight than the traditional industrial robots. This allows much easier move and set-up from task to task and the manipulator can be for instance installed on top of a small sized mobile robot. However the payload of the collaborative robots is usually relatively small (< 20 kg) making them practical only on medium size applications.
- Easy to program. Most of the collaborative robots come with a touch sensitive teach pendant with simple UI to program and move the robot. In addition, many of manufactures provide tablet or smartphone application to do simple tasks with robot.
- In contrast to traditional industry robots and fully automated robot applications the collaborative robots are meant to aid and assist, not to replace the human worker.
- With a lightweight structure, simple design and ease of programming, the collaborative robots are less expensive and easier to maintain than the traditional industrial robots.

The first generation of so called lightweight robots (LWR) was presented by the Institute of Robotics and Mechatronics at German Aerospace Center in the 1990s [58]. The origin of the project was in space robotic research where the target was to develop a robotic arm for astronauts due to their need for a light and flexible multi-joint actuator. Although the development was pushed by space robotic requirements, the lightweight robot had its breakthrough with terrestrial applications. The released product for industrial applications had seven degrees of freedom achieving the same dexterity as the human arm and having the total system-weight less than 20 kg. Today many traditional robot manufacturer have their own collaborative robot in their product catalog (see Fig. 4.2). For instance ABB (Swiss-Swedish) has developed a dual arm robot system with integrated vision sensors in both grippers aimed for small parts assembly. Collaborative robots by Universal Robots (Denmark) have been extensively used in many different research groups due to their relatively affordable price and easy-to-use interface. For less expertised fields, such as health care and education, Rethink Robotics (United States) released their friendly looking collaborative robot called Sawyer having a large display to indicate the current robot state using human-like expressions.

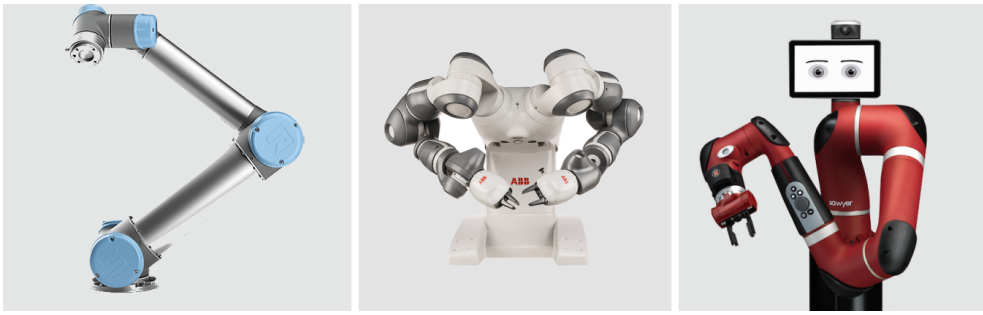


Figure 4.2 Collaborative robots from different manufacturers (from left to right): UR5 from Universal Robot family, ABB YuMi and Rethink Robotics Sawyer.

4.2 Safe HRC

In order to create collaborative applications where a person feels safe while working together with the robot, it is necessary to understand what constitute safety hazards and define common safety requirements and strategies. In simple terms, unintentional and unwanted contact between the robot and human has to be prevented using external safety devices such as sensors and mechanical switches. If the task requires physical contact or it is impossible to exclude the chance of collision, then velocities and forces exerted upon the human must remain below thresholds for physical discomfort or injury.

4.2.1 Safety standards and criteria

The International Organization for Standardization (ISO) is the main institution defining and releasing documents that describe the best practices how to maintain safety during interaction between humans and robots. In 2006 the ISO 10218 document was released and it was the first step of defining safety requirements for industrial robots. The document describes different situations and corresponding safety measures that has to be taken into account. For instance the operator has to stay outside the robot cell while the robot is running on automatic mode, but during robot programming, the operator may stay close to a slow moving robot (< 250 mm/s) when the hold to run control is applied. The guideline was revised in 2011 to give more possibilities for robot system integrators to design safe and productive robot

cell. The standard consists of two different parts, ISO 10218-1:2011 [62] and ISO 10218-2:2011 [63], which were targeted to robot manufactures and robot system integrators, respectively. Technical specification ISO/TS 15066 [65] has been recently published to address the safety requirements of collaborative industrial robot systems and supplements the requirements in ISO 10218–1 and ISO 10218–2. In addition, it also introduced completely new requirements such as pain threshold map that defines the maximum forces that may occur in collaborative mode on different human body parts.

As a summary, the standard defines four main techniques for collaborative operation for collaborative applications: *safety-rated monitored stop*, *hand-guiding operation*, *speed and separation monitoring* and *power and force limiting*.

- **Safety-rated monitored stop (SMS):** Robot can operate autonomously when no one else is inside the collaborative workspace and the robot is immediately halted if stop condition is met e.g. human co-worker violates the safety area. Restarting the robot after safety-rated monitored stop can be automatic if there are no persons at the vicinity of the robot. Potential applications are direct part loading or unloading to end-effector and work-in-process inspections.
- **Hand-guiding operation (HG):** Typical in lift assisted tasks, where the human operator uses hand-operated device to transmit motion commands to the robot. A safety-rated monitored stop must be issued before activating the hand-guidance.
- **Speed and separation monitoring (SSM):** The robot system designed to maintain safe separation distance during a collaborative task. The robot velocity can be automatically adjusted based on the operator and robot relative distance. Practical in simultaneous tasks where the human and robot operate inside the same work space.
- **Power and force liming (PF):** At the collaborative workspace the robot applies velocities and forces, which are not harmful to a person. Physical contact between the robot system and the human co-worker can happen intentionally or unintentionally. If any parameter limits are exceeded, a safety-rated stop is issued. Suited for applications where the human has to be frequently right next to the robot.

4.2.2 Safety strategies

The ISO/TS documents have been criticized to be more like guidelines than a clear definition of HRC applications and safety requirements. Therefore several authors have provided their own design guidelines and concepts corresponding to next generation manufacturing and aligned with today's safety standards. In [8, 142] the collaboration was classified to different interaction levels and for each level, different type of safety functions were developed, linked and analyzed. Both of the authors identify four interaction levels where in the bottom (*Level 1*), the human and robot work in the same space but have their own tasks. The shared workspace is fenceless and the workspace is divided spatially into two virtual zones: human and robot zone. The human zone is static where as the robot zone can be configured either to be dynamic (calculated based on the robot motion) or static. In *Level 2* the human and robot share the task but without physical interaction. There is no direct contact and the robot can only move towards the human for instance while bringing a work piece to a predefined position. In addition, the robot can hold the component while the human operator does the assembly. The third level (*Level 3*) is similar to Level 2 where no physical interaction is allowed. However the robot is now allowed to grasp objects directly from the human hands or other way around. The level requires special zone for the handing-over tasks where a reactive motion controller has to implemented in order to adapt the human hand movements. In the final level (*Level 4*) the robot and human share the workspace and task with physical interaction. For instance human force guides the end-effector while the robot at the same time adds upward force to make the carried object lightweight. In their taxonomy, SMS is required in each interaction level to stop the robot in a case of safety violations. Control of other robot parameters (velocity and force) as well as image processing algorithms for detecting different human features (gestures, facial expressions etc.) are required in the higher levels of interaction

Lasota et. al [77] provided a comprehensive survey of existing safety strategies in HRC and divided the methods into four different direction: *Safety Through Control*, *Safety Through Motion Planning*, *Safety Through Prediction* and *Safety Through Consideration of Psychological Factors*. Safety through control is currently one of the most active research field in HRC safety where the collision is prevented for instance by stopping or slowing down the robot without long-term planning algorithms [51].

In other studies [93, 94], the safety issue is discussed from the perspective when the collision between the human and robot cannot be necessarily avoided. The authors summarized three different strategies for safety: *crash safety* (controlled collision using power/force control), *active safety* (external sensors for collision prediction) and *adaptive safety* (applying corrective actions that lead to collision avoidance).

4.2.3 Vision-based safety systems

Vision-based methods are one of the most popular and direct ways of gaining surrounding information such as environmental geometry and human intention for collision detection. Among vision-based methods, the efficiency of collision detection has been the motivation for many researchers. One of the earliest approaches in industrial environments was to use volumetric virtual zones, where a movement inside a certain zone would signal an emergency stop or slow down the robot. SafetyEYE (Pilz)[3] and SafeMove (ABB)[4] are few standardized and commercialized vision-based safety systems that use an external tracking system to monitor movement inside predefined safety regions. In contrast, the authors [135, 138] presented an approach where the regions can be updated during run-time. In [139] a dynamic robot working area was projected on a flat table by a standard digital light processing (DLP) projector and safety violations were detected by multiple RGB cameras that inspect geometric distortions of the projected line due to depth changes. In [43] the requirement for flat display medium was lifted by the proposed projection mapping system (single RGB camera and DLP projector) that can take into account the object geometric structure.

Depth sensing has become a popular and efficient approach to monitor the shared environment and to prevent collision between the robot and an unknown object (e.g. human operator). In most of the approaches a virtual 3D model of the robot is generated and tracked at run-time while real measurements of the human operator from the depth sensor are used to calculate the distance between robot and human body parts. Depth sensing is then combined with reactive and safety-oriented motion planning that guides the manipulator to prevent collisions [22, 39, 107]. Moreover, recent research [16, 71] have discussed an efficient and probabilistic implementa-

[3] https://www.pilz.com/en-INT/eshop/00106002207042/SafetyEYE-Safe-camera-system
[4] https://new.abb.com/products/robotics/controllers/irc5/irc5-options/safemove-2

tion of SSM as dictated by ISO/TS, where the safety system has dynamic control of the safety distance between the robot and human operator such that it complies with the minimum safety requirements. For a practical application these methods have to be extended to multi-sensor systems where the possibility of having occluded points is removed [36]. Current consumer-grade RGB-D sensors can deliver up to several million point measurements in a second which requires substantial computational power. For real-time interaction more complex implementations have been proposed such as GPU-based processing [19] and efficient data-structures [146].

Traditional machine learning techniques have been used in HRC for long-term planning and human intention recognition in industrial context. In [134] the authors proposed an efficient and safe motion planning model, that is trained on observed human motion data. The model consists of three different classifiers that together predict the next most likely human action and incorporate this to the motion planning. In [87] human actions during an assembly were categorized through the use of Gaussian Mixture Models (GMMs) and Gaussian Model Regression (GMR). Liu and Wang [84] used Hidden Markov Model (HMM) to classify human intention analysis and further as an input for assistive robot motion planning. Due to the success of artificial neural networks in vision and audio related applications, researchers have started to experiment them on safe HRC. In [108] a vision-based neural network monitoring system is proposed for locating the human operator and ensuring a minimum safe distance between the co-workers. In parallel, deep models have been proposed for human hand and body posture recognition [82] and intention recognition in manufacturing tasks [141]. However, most of the approaches assume all human actions to be from a known observation set and are not designed to work for unseen actions.

4.3 Safety through robot control

One common method to avoid collision between the human and robot is to utilize robot motion parameters and stop, slow down or guide the robot motion away from the human. Safety though control can be further divided to two main categories, *Speed and Separation Monitoring* and *Potential Field Methods*, which do not require planners or complex predictions systems that can be difficult to implement to work reliably in real-time [77].

4.3.1 Speed and separation monitoring

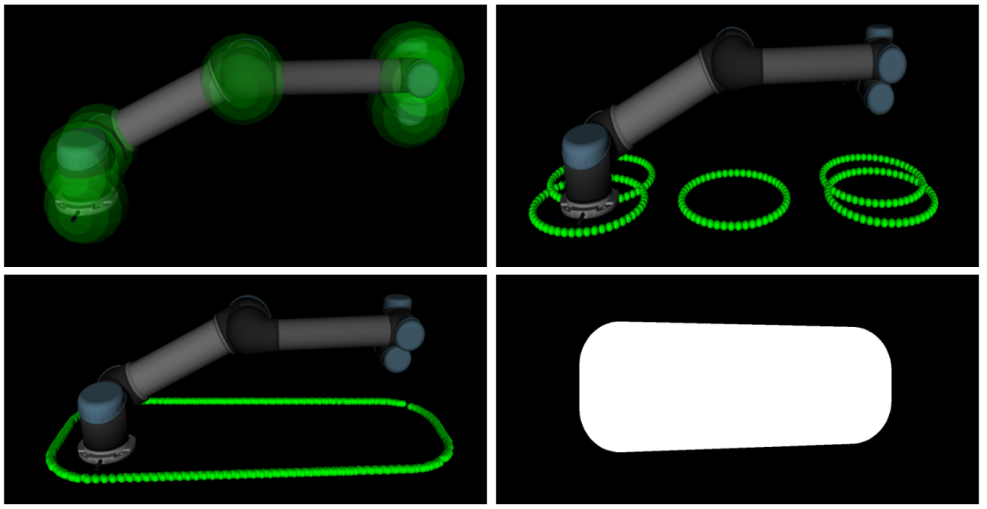


Figure 4.3 Steps for computing a dynamic safety zone. Top left: the computed 3D control points ϕ_i. Top-right: the control point xy-coordinates map directly to the robot frame xy-plane and are converted to regions by plotting circles of radius r. Bottom-left: [47] algorithm provides a convex hull around the circles. Bottom-right: convex hull converted to a 2D binary image.

Minimum protective distance. Stopping the robot through the use of safety zones that change dynamically based on the robot position has been studied in several recent studies [**P3**, 138] These methods particularly enable real-time monitoring and securing of the minimum protective distance between a robot and an operator. In [**P3**] a minimum safety hull encapsulating the robot is generated via a virtual robot 3D model. More specifically a set of N control points $\mathscr{R} = \{\vec{\phi} : \vec{\phi} \in \mathbb{R}^3\}$ is generated and tracked during run-time using the robot kinematics and robot joint values. The point locations are selected on the robot arm so that they cover the extreme parts of the robot. During run time, the control points are projected to robot xy-plane and simplified to 2D circles. The radius of the circles is controlled by the free parameter ω that should be selected based on the robot dimensions. Efficient convex hull algorithm of Graham et al. [47] is used to encapsulate all the circles and ultimately forming the hull of the safety zone. For faster inference, the safety hull is finally transformed to a 2D binary mask representation (see Fig. 4.3) in which the hull does

not have lower or upper bounds in z-direction. In [**P4**] an important extension to the depth- and zone-based monitoring is proposed where the virtual safety hull is extended over the carried object. In such a case a new set of control points is created using known dimensions of the object and the robot current configuration. Using fast binary operations, the hull of the object and robot is connected. At run-time, anomalies inside the hull can be detected in real-time using a depth sensor (see Sec. 5.2).

In addition to zone-based monitoring, lots of research work has focused on direct distance-based methods, where the smallest distance between the robot and any sensor detection $\mathcal{O}_i$, is determined by searching for the overall minimum

$$S = \min_{\forall i,j} \left\| \vec{\phi}_j - \mathcal{O}_i \right\| \quad , \tag{4.1}$$

where $\vec{\phi}_j$ is the predefined control point on the robot. In this scenario the obstacle (general objects, human limbs etc.) locations has to be known reliable and in real-time. Two methods to detect human and limbs have been widely considered in literature: vision-based methods and inertial sensor-based methods using a special suit for motion capture [140]. The latter approach may not be considered as a realistic solution for real applications because of the need of wearing a special uniform with sensing devices and insufficient detection of movement in the environment around the human. Recent vision-based methods [8, 16, 71, 90] have focused on identifying and implementing methods that comply with the third collaborative scenario (SSM) of ISO/TS that defines the protective separation distance S_p at the current time t_0 using six variables:

$$S_p(t_0) = O_m + R_r + R_s + Y_d + Y_a + O_b \quad . \tag{4.2}$$

S_p is expressed as the sum of contributions by robot motion (reaction distance R_r and stopping distance R_s after stop command has been signaled), operator motion (O_m) and uncertainties related to sensor systems (Y_d) and robot attributes (Y_a). In ISO 13857 [64], minimum separation distance for different body parts are defined and the information is included in O_b. Based on the standard, the robot is halted if S_p is less than a predefined threshold value τ_S.

Speed monitoring. Forward and inverse kinematics relate joint positions to end effector position and orientation and vice verse. Determinating the kinematics of the robot is the basis for any robotic based manipulation, where the end effector has to be moved to desired position and orientation. However in many cases we are interested about the velocity relationship, i.e. relating the end effector linear $\vec{v} = (v_x, v_y, v_z)^T$ and angular $\vec{\omega} = (\omega_x, \omega_y, \omega_z)^T$ velocities along the main axes to the joint velocities. This is especially useful feature in HRC where the robot's end effector velocity (or any other point on the robot manipulator) has to be slowed down due to the human present in close proximity [94, 106, 134]. The velocity relationship between the space of a Cartesian position and orientations of the end effector and the space of joint positions is determinated by the *Jacobian matrix* (or simply *Jacobian*). The Jacobian is a $6 \times n$ matrix where n is the number of joints in the robot manipulator

$$\boldsymbol{J} = [\vec{j}_1 \vec{j}_2 \dots \vec{j}_n] \ . \tag{4.3}$$

For joint constraining the motion between two bodies to pure rotation along a single axis (i.e. revolute joint) the i-th column $\vec{j}_i$ in the Jacobian matrix is

$$\vec{j}_i = \begin{bmatrix} \vec{z}^0_{i-1} \times (\vec{o}^0_n - \vec{o}^0_{i-1}) \\ \vec{z}^0_{i-1} \end{bmatrix} \ , \tag{4.4}$$

where $\vec{o}_n$ is the 3D point on the manipulator respect to the base frame for which the Jacobian is calculated. Coordinate point of each revolute joint $\vec{o}_{i-1}$ and their corresponding axis vector $\vec{z}_{i-1}$ are defined relative to the base frame and calculated from the transformation matrix $\boldsymbol{T}(\vec{q})^0_{i-1} = [\boldsymbol{R}^0_{i-1} | \vec{t}^0_{i-1}]$ as

$$\vec{z}_{i-1} = \boldsymbol{R}^0_{i-1} \vec{k}, \quad \vec{o}^0_{i-1} = \vec{t}^0_{i-1} \ , \tag{4.5}$$

where $\vec{k} = (0, 0, 1)^T$. For joint which constrains motion between two bodies to pure linear motion along a single axis (i.e. prismatic joint) the i-th column is:

$$\vec{j}_i = \begin{bmatrix} \vec{z_{i-1}} \\ \vec{0} \end{bmatrix} \tag{4.6}$$

Determinating the Jacobian matrix for any manipulator using the above formulas is straightforward since all the needed quantities are already available if the forward kinematics (transformation chain up to link n) have been worked out.

With the Jacobian we can define the mapping

$$\dot{\vec{X}} = \boldsymbol{J}(\vec{q}) \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix} , \tag{4.7}$$

between the end effector twist $\dot{\vec{X}} = (\vec{v}, \vec{w})^T$ and the vector $\dot{\vec{q}}$ of joint velocities. Each of the rows in the matrix $\boldsymbol{J}$ specify the influence of a specific joint to the end effector velocities. For instance with a robot having only few active joints, the matrix might have rows consisting of zeros meaning the corresponding velocity components in the end effector twist vector cannot be controlled.

Previously, we transformed the joint velocities to end effector twist. More interestingly we can calculate required joint angle velocities for a desired end effector twist

$$\begin{bmatrix} \dot{\vec{q}}_1 \\ \dot{\vec{q}}_2 \\ \vdots \\ \dot{\vec{q}}_n \end{bmatrix} = \boldsymbol{J}(\vec{q})^{+} \dot{\vec{X}} , \tag{4.8}$$

where $\boldsymbol{J}^{+}$ is the pseudo-inverse of the Jacobian.

4.3.2 Potential field methods

Another popular collision avoidance technique using robot control is the *potential field* approach that uses an artificial potential field to guide the robot manipulator [70]. The environment consists of forces that either pull or push the robot. In particular, the robot has a goal position (e.g. pose of the manipulator) in the environment that is dragging the robot while at the same time, the obstacles generate repulsive forces that pushes the end effector away from them. In contrast to speed

and separation monitoring the potential field approach allows more complex safety features where the robot trajectory can be changed at run-time based on dynamic workspace factors.

In general, the potential field U consists of two components that have influence on a 3D point $\vec{\phi}$ in the Cartesian space, attractive potential field $U_{\text{att}}(\vec{\phi})$ and repulsive potential field $U_{\text{rep}}(\vec{\phi})$

$$U(\vec{\phi}) = U_{\text{att}}(\vec{\phi}) + U_{\text{rep}}(\vec{\phi}) \ . \tag{4.9}$$

One of the easiest ways to guide the robot to a goal position without collisions is to use gradient descent, where we search the global minimum of U by following the negative gradient of the formula

$$F = -\nabla U(\vec{\phi}) = -\nabla U_{\text{att}}(\vec{\phi}) - \nabla U_{\text{rep}}(\vec{\phi}) \ . \tag{4.10}$$

The attractive potential field guides the robot to goal position $\vec{\phi}_{\text{goal}}$. The simplest choice of the potential field U_{att} is the quadratic field function

$$U_{\text{att}}(\vec{\phi}) = \frac{1}{2}\gamma ||\vec{\phi} - \vec{\phi}_{\text{goal}}||^2 \ , \tag{4.11}$$

where γ is the gain parameter. The magnitude of the attractive gradient linearly decreases while the system comes closer to the goal position

$$-\nabla U_{\text{att}}(\vec{\phi}) = \gamma(\vec{\phi} - \vec{\phi}_{\text{goal}}) \ . \tag{4.12}$$

However, it might be desirable to have a distance function that grows slowly to avoid huge velocities far from the goal position. In practice, a combined attractive field function is used for instance by applying a quadratic potential near the goal and a conical potential farther away.

The main goal of the repulsive force field is to repel the robot from obstacles and thus ensure that the robot does not collide with the obstacles in the workspace. If the robot is far away from the obstacles, the obstacles should have little or no effect on the current motion of the robot. In [70] this is achieved by a formula where the repulsive force goes towards infinity near the obstacle boundary and is zero after the

distance is greater than a predefined threshold value ρ_0

$$U_{\text{rep}}(\vec{\phi}) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(\vec{\phi})} - \frac{1}{\rho_0}\right)^2 & \text{if } \rho(\vec{\phi}) \geq \rho_0 \\ 0 & \text{if } \rho(\vec{\phi}) < \rho_0 \end{cases} . \tag{4.13}$$

In the equation, $\rho(\vec{\phi})$ is the shortest distance from $\vec{\phi}$ to a obstacle boundary and η is a scalar gain coefficient that determines the influence of the repulsive field. The robot is driven away from the obstacle by following the negative gradient of the repulsive field

$$-\nabla U_{\text{rep}}(\vec{\phi}) = \begin{cases} \eta\left(\frac{1}{\rho(\vec{\phi})} - \frac{1}{\rho_0}\right)^2 \frac{1}{\rho^2(\vec{\phi})} \nabla\rho(\vec{\phi}) & \text{if } \rho(\vec{\phi}) \geq \rho_0 \\ 0 & \text{if } \rho(\vec{\phi}) < \rho_0 \end{cases} , \tag{4.14}$$

where the partial derivative to the nearest obstacle surface point $\vec{b}$ at $\vec{\phi}$ is

$$\nabla\rho(\vec{\phi}) = \frac{\vec{\phi} - \vec{b}}{||\vec{\phi} - \vec{b}||} . \tag{4.15}$$

When potential fields are utilized on robot manipulators, we define set of 3D points $\vec{\phi}_i$ that are equally distributed on the robot body. In general, the control points $(\vec{\phi}_1, \vec{\phi}_2 \ldots \vec{\phi}_{N-1})$ are assigned one point per link. Their position can be fixed for instance at the mass center point of the links or we might want to search the closest point respect to the obstacle along an arm segment in which case the control point location can be anywhere between consecutive joint axis. The final point $\vec{\phi}_N$ is located at the end effector. The attractive potential field U_{att} is calculated only based on the $\vec{\phi}_N$ location where as the repulsive potential U_{rep} acts on the whole set of $(\vec{\phi}_1, \vec{\phi}_2 \ldots \vec{\phi}_N)$. The total potential field is

$$F(\vec{q}) = -\boldsymbol{J}(\vec{q})_0^N \nabla U_{\text{att}}(\vec{\phi}_N) - \sum_{n=1}^{N} \boldsymbol{J}(\vec{q})_0^i \nabla U_{\text{rep}}(\vec{\phi}_i) , \tag{4.16}$$

where $\boldsymbol{J}_0^i$ is the Jacobian for the point of interest $(\vec{\phi}_i)$. In addition, the resulting force vector can be directly considered as a joint velocity vector $\dot{\vec{q}}$ and fed to the low-level controller of the robot. This enables fast realization of motion corrections but does not consider robot dynamics which might lead to jerky robot motion in some

cases. In HRC, artificial field approaches have been utilized extensively as the main safety strategy [22, 39, 107] or as a part of bigger collection policies [81]. The main drawback of this approach is the presence of local minima, which can trap the robot before reaching its goal.

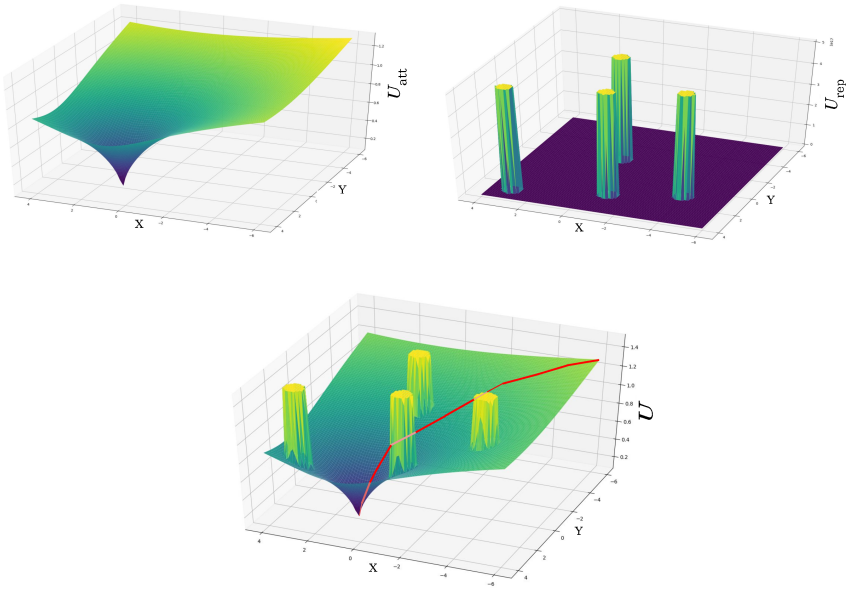


Figure 4.4 The robot can navigate though an environment as a particle under the influence of an artificial potential field. Top left: the robot is attracted by a single goal position that creates an attractive potential field U_{att} to guide the robot. Top right: Multiple spherical obstacle in the environment causing a repulsive force U_{rep} on the robot. The repelling force goes towards infinity at the object boundary. Bottom: The total potential field U is the sum of U_{att} and U_{rep}. The robot is guided by the negative gradient $-\nabla U$ indicating the most promising local direction of motion.

4.4 AR-based operator support system

In human interaction, the ability to understand each others through various signals is critical for successful collaboration within human teams. In HRC, the ability of understanding the internal state and intentions of a robot is crucial for safe and efficient collaboration. This is common in situations where the human enters the robot work environment for a brief inspection as well as in situations where the robot and human share the work space for longer periods of time. In addition, intuitive UI

is one of the most important aspects when adopting a complex robot system to a new environment where the system operators do not have the specialized knowledge about hardware or robot programming.

In robotics different communication modalities have been used for seamless interaction such as gestures, voice commands and graphical UIs. One major advantage of using voice commands is that it frees the hands of the operator, allowing him to control the robot while performing a shared task on his own. However, in manufacturing or other noisy environments the voice commands should not be used alone but combined with other modalities to overcome the possibility of faded spoken action. In addition to speech, gestures are also a natural way of communication for humans to exchange information. The research study in [88] introduces an intuitive system for robot programming where gestures appear as one of the methods to command the robot in a industrial environment. The gestures presented in the work are fixed and defined in advance. However, due to the variability in the execution of different hand gestures and poor lightning condition, the recognition of the gestures might be difficult. In the following sections we are mainly focusing on graphical UI that can be established and monitored by vision-based techniques. In particular, these methods can be used as two-way communication channel between the robot and human.

Advances in display and vision technologies have created new interaction modalities that enable informative and real-time communication in shared workspaces. In robotics, various different signaling techniques have been proposed during the years and one common way is to project 2D information to a table or floor [48]. One of the earliest approaches to create a communication interface between robot and human was introduced in [117]. The paper presents a system that visually tracks the operator's pointing hand and projects a mark at the indicated position using an LCD projector. The marker is then utilized by the robot in a pick-and-place task. More recently, Vogel et. al [139] used a projector to create a 2D display with virtual interaction buttons and textual description that allow intuitive communication. In another recent work [3, 43] the authors proposed a projector-based display for HRC in industrial car door assembly. In contrast to other projector-based works, the system can display visual cues on complex surfaces. User studies of the system against two baselines, a monitor display and simple text descriptions, showed clear improvements in terms of effectiveness and user satisfaction.

Wearable AR such as head-mounted displays (HMD) and stereoscopic glasses have recently gained momentum as well. The earliest versions of wearable AR devices were typically considered bulky and ergonomically uncomfortable when used over long periods of time [5]. In addition, each of the human participants in the collaborative task is required to wear the physical device. However, 2D displays can only provide limited expression power and can be more easily interfered, for instance, due to direct sunlight or obstructing obstacles. In [111] a HMD was used for communicating the robot intention for human co-worker and the method effectiveness against a 2D display was verified in a simple toy task. Huy et al. [61] demonstrated the use of HMD in an outdoor mobile application where a projector system cannot be used. Elsdon and Demiris [34] introduced a handheld spray robot where the control of the spraying was shared between human and robot. In [46] the authors combined two wearable AR-gear, a head-mounted display and a smartwatch, for supporting operators in shared industrial workplaces.

4.5 Summary

The current market in industrial manufacturing requests more flexible and multi-purpose assembly stations to solve the existing challenges in assembly lines. Collaboration between humans and robots is seen as a promising step toward more productive manufacturing floors while decreasing production costs [91, 140]. Today, most of the existing workcells consist of isolated robots that use static strategies to execute a task. This does not support well the dynamic nature of HRC where the human and robot are working side-by-side on a common task. In order to achieve high performance production in HRC, the robot system has to be fundamentally safe for the human operator, communication between the two co-workers has to be intuitive, and the system should be easy to set up.

In the past, research on HRC has been active and a number of different safety techniques and interaction modalities have been proposed. Among various techniques, vision- and robot control-based methods have proven to be efficient and reliable solutions for collision detection. In addition, visual UIs capable of augmenting the workplace with graphical information have gained positive momentum due to their ability to instruct human operator in complex tasks.

5 APPLICATION OF SAFE HRC

5.1 Introduction

In the previous chapter HRC in industrial manufacturing was briefly discussed. In HRC the most crucial task is to ensure the safety of the human co-worker and in manufacturing settings this requires special attention since heavy robots and payloads can lead to potentially dangerous situations. In addition, to be effective and efficient, the HRC safety systems should be affordable and easy-to-install.

In this chapter, we present a depth sensor and AR-based safety model for a shared workspace for collaborative manufacturing. The chapter summarizes the work that has been conducted in the publications [**P3**, **P4**, **P5**]. For the model, we adopt the concept of workspace safety zones by Bwidi et. al [7, 8]. For interaction, a UI implemented on two different hardware, *projector-camera* and *HoloLens*, is introduced. On the model four different zones are defined: *robot*, *danger*, *cooperation* and *human*. The model contains functionality to detect danger zone violations and update changes in the shared workspace automatically by the robot or manually through human verification. At the end of the chapter, experiments on the safety model in two different assembly tasks and against baseline methods are described. Finally, a summary of the qualitative and quantitative results from the experiments is provided.

5.2 Shared workspace model

The shared workspace S is surveilled by one or multiple depth sensors that actively monitor operations inside the workspace. A depth sensor can be modeled as a simple pinhole camera and it is parameterized by two matrices: the intrinsic camera matrix $\boldsymbol{K}$, modeling the projection of a Cartesian point to a image plane, and the extrinsic

camera matrix $\boldsymbol{T} = [\boldsymbol{R}|\vec{t}]$, describing the pose of the camera in the robot coordinate system. Both of the matrices can be acquired through standard calibration procedures.

After all the necessary calibration is done the workspace model $\boldsymbol{I}_S$ is created and updated during run-time. The model can be created offline by taking multiple images of the shared workspace and registering all the different views to robot base frame. Each of the Cartesian points $\vec{p} \in \mathbb{R}^3$ from a sensor j is projected to workspace model $\boldsymbol{I}_S = \{\vec{x}_i\}_{i=1}^{w \times h}$ as

$$\vec{x} = \boldsymbol{T}_{proj}\left(\boldsymbol{N}^{-1}\left(\boldsymbol{R}_j \boldsymbol{K}_j^{-1} \vec{p} + \vec{t}_j\right)\right), \tag{5.1}$$

where $\boldsymbol{N}^{-1}$ is the inverse coordinate transformation and $\boldsymbol{T}_{proj}$ is the projective transformation, scaling and translating the model origin from the robot frame to the model reference frame. During run-time new measurements from the depth sensor(s) are projected to the same space as the workspace model. Above transformations define a simple and efficient representation of the workspace that enable real-time safety monitoring. All the HRC zones explained in the following are defined on the workspace model.

5.2.1 HRC Zones

The model divides shared workspace into four different zones: robot zone $\boldsymbol{Z}_r$, danger zone $\boldsymbol{Z}_d$, cooperation zone $\boldsymbol{Z}_c$ and human zone $\boldsymbol{Z}_h$. Operations on the zones are done efficiently using bitwise operations $\cup$ (or) and $\setminus$ (substraction). The zones are initialized using the procedure described in Section 4.3.1 and modelled as binary masks defined on the $w \times h$ sized workspace model.

Robot zone. The robot zone $\boldsymbol{Z}_r$ is a dynamic hull around the robot that encapsulates all the parts of the robot. The main target of $\boldsymbol{Z}_r$ is to automatically update the workspace model based on the robot actions. For instance, if the robot lifts a work part and installs it on a target object, then all the depth changes in the workspace are automatically updated to the workspace model $\boldsymbol{I}_S$. All the other HRC zones used in the model are generated based on the robot zone. The robot zone $\boldsymbol{Z}_r$ is generated using the mask operator $M_r(\cdot)$ creating the hull based on tracked control points

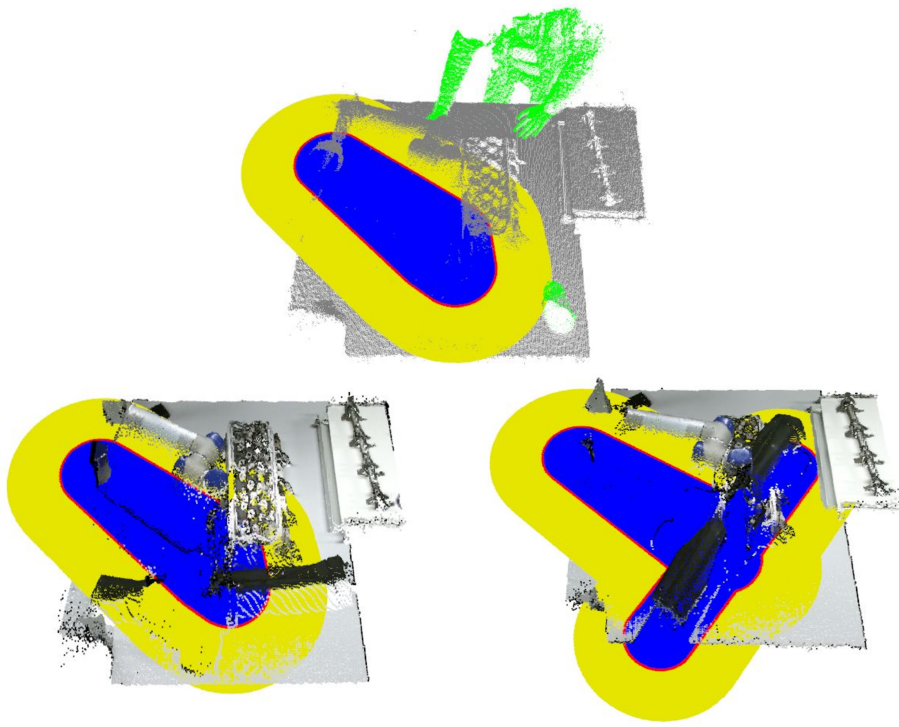


Figure 5.1 A shared workspace S is modelled as a depth map image I_S and divided to four HRC zones. The robot zone Z_r (blue) is dynamically updated and subtracted from I_S to generate the human zone Z_h (gray). The two zones are separated by the danger zone Z_d (red) which is monitored for safety violations. The hull of the danger zone is the collaborative zone Z_c (yellow), where the speed of the robot is reduced. Changes in Z_h are recorded to binary masks R (green). Manipulated objects are automatically added to HRC zones (bottom right).

covering all the extreme parts of the robot (more details in 4.3.1)

$$Z_r = M_r(\omega) \ . \tag{5.2}$$

The free parameter ω is robot specific and depends for instance on the size of the robot.

Danger zone. The danger zone Z_d is the contour of Z_r and essentially separates the human and robot to their respective work spaces. The main task of the zone is to ensure that human or any other unregistered object can not enter inside the

robot zone. If the safety model registers any anomalies inside the zone, the robot is directly stopped and the robot must be restarted from the UI (see Sec. 5.3.2). The zone is constructed using the fast binary operators and by adding the danger margin ω_d

$$Z_d = M_r(\omega + \Delta\omega_d) \setminus Z_r \ . \tag{5.3}$$

Cooperation zone. Inside the cooperation zone the human and robot share the task but the cooperation is limited. The primary target of the zone is to reduce the speed of the robot while it is approaching or working in close proximity the human operator. In a typical scenario the robot brings and holds a work part for the human operator while he/she fastens the part on the target object. The cooperation zone is created adding the margin ω_c to the binary mask operator $M_r(\cdot)$

$$Z_c = M_r(\omega + \Delta\omega_c) \setminus (Z_d \cup Z_r) \ , \tag{5.4}$$

where $\Delta\omega_c > \Delta\omega_d$.

Human zone. Inside the human zone the operator can move freely and none of the robot motion parameters require controlling i.e. the robot can move at the maximum allowed speed. The human zone is calculated by subtracting all the other zones from the workspace model

$$Z_h = I_S \setminus (Z_r \cup Z_d \cup Z_c) \ . \tag{5.5}$$

Extending the HRC zones over carried object. Collaborative robots may be designed inherently more safe than the traditional industrial robots i.e. they are much lighter and might have internal joint torque sensors for reducing the damage during evident collision. However, the design of the robot tool and work parts can significantly impair the safety during the collaboration. For instance, the robotic task (see 5.4.1) includes work parts that have sharp edges and are moderately heavy ($>$ 4kg). Thus, an important extension of the model is that the known objects that the robot manipulates are added to the HRC zones. This guarantees that the robot does not accidentally hit the operator while the object is being carried. The zones

are extended based on the formulas

$$\mathbf{Z}_r = M_r(\omega) \cup M_{obj}(\omega) \tag{5.6}$$

$$\mathbf{Z}_d = M_r(\omega + \Delta\omega_d) \cup M_{obj}(\omega + \Delta\omega_d) \setminus \mathbf{Z}_r \tag{5.7}$$

$$\mathbf{Z}_c = M_r(\omega + \Delta\omega_c) \cup M_{obj}(\omega + \Delta\omega_c) \setminus (\mathbf{Z}_d \cup \mathbf{Z}_r) \ . \tag{5.8}$$

5.2.2 Safety monitoring

In HRC the first priority is to ensure that the human and robot do not collide to each other. Thus the main principle of the shared workspace model is to monitor depth changes inside the danger zone $\mathbf{Z}_d$. Any changes inside the zone results in immediate halt of the robot. Our depth-based model in the robot frame $\boldsymbol{I}_S$ provides fast computation since the change detection is computed as a fast subtraction operation

$$\boldsymbol{I}_\Delta = \|\boldsymbol{I}_S - \boldsymbol{I}\| \ . \tag{5.9}$$

where $\boldsymbol{I}$ is the most recent depth data transferred to same space as our workspace model. The difference bins (pixels) are further processed by Euclidean clustering [113] to remove spurious bins due to noisy sensor measurements.

Finally, the safety operation depends on which zone a change is detected:

$$\boldsymbol{R} = \mathbf{0}, \forall \vec{x} \mid I_\Delta(\vec{x}) \geq \tau \begin{cases} \text{if } \vec{x} \in \mathbf{Z}_d & \textit{HALT} \\ \text{if } \vec{x} \in \mathbf{Z}_c & \textit{SLOW DOWN} \\ \text{if } \vec{x} \in \mathbf{Z}_r & \boldsymbol{I}_S(\vec{x}) = \boldsymbol{I}(\vec{x}) \\ \text{if } \vec{x} \in \mathbf{Z}_h & \boldsymbol{R}(\vec{x}) = 1 \end{cases} , \tag{5.10}$$

where τ is the depth threshold. In the first case, the change has occurred in the danger zone $\mathbf{Z}_d$ and therefore the robot must be immediately halted to avoid collision. For maximum safety this processing stage must be executed first and must test all pixels $\vec{x}$ before the next stages.

In the second case, the human works in close proximity of the robot or the robot is reaching the human for handing-over task, for instance a tool or work part transfer. In this case, the robot speed is reduced and the human feels more comfortable and has more time to react with respect to the robot motion.

In the third case, the change has occurred in the robot working zone Z_r and is therefore caused by the robot itself by moving and/or manipulating objects and therefore the workspace model $\boldsymbol{I}_S$ can be safely updated.

In the last case, the change has occurred in the human safety zone Z_h and we create the mask $\boldsymbol{R}$ that represents the changed bins. It is noteworthy that the mask is recreated for every new frame from the sensor to allow temporal changes, but it does not affect robot operation. The robot can continue operation normally, but if its danger zone intersects with any 1-bin in $\boldsymbol{R}$, then these locations must be verified from the human co-worker via UI. If the bins are verified, then these values are updated to the workspace model $\boldsymbol{I}_S$ and operation continues normally. Note that our model does not verify each bin separately, but a spatially connected region of changed bins. This operation allows a shared workspace and arbitrary changes in the workspace which do occur away from the danger zone.

5.3 Setup

5.3.1 Robot platform

For experimenting and demonstrating the proposed safety model, a mobile platform was created that can be easily moved and deployed to new locations. The platform consists of a wheeled table and metallic columns in the table sides that hold a projector and RGB-D sensor. The wide-angle 3LCD projector is installed in top of the table pointing downwards to the workspace area. The wide angle lens replaces the tilted mirror that was used in the previous robot platform [**P3**] to expand the projection area. The projector outputs a 1920×1080 color image with 50 Hz frame rate. Kinect v2 was installed next to the projector capturing the whole workspace area. The platform is designed to work with a head-mounted AR display by attaching calibration marker in fixed location on the table which position respect to the robot is known.

The UR5 robot from Universal Robot family is installed on the table with the OnRobot RG2 gripper. The robot is a collaborative arm with 6 separate joints, a carrying capacity of 5 kg and a spherical operational radius of 850 mm. The gripper is especially designed for Universal Robot and has a long stroke allowing the gripper to handle a variety of object sizes. However the gripper has relatively low gripping

force (40 N), making it less practical for handling heavy objects. For the gripper custom made fingers were 3D printed and reinforced by nylon and carbon fiber mixed filament. The main design requirement for the fingers was to handle variety of object shapes in order to avoid additional tool chain during the operations.

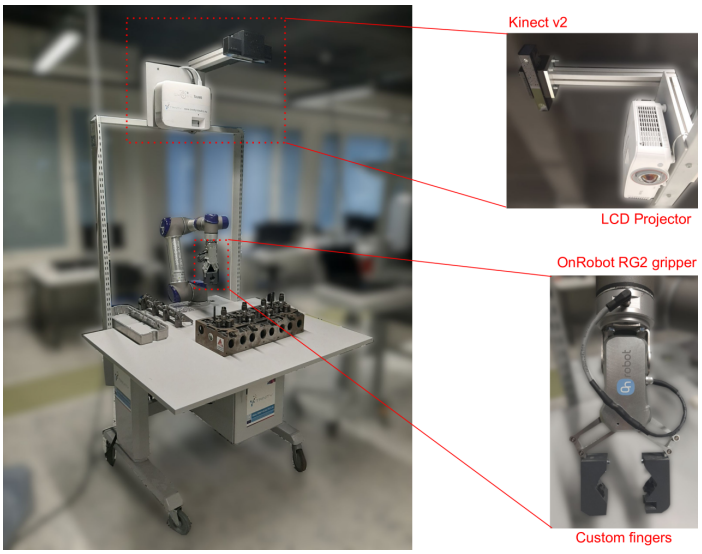


Figure 5.2 A movable robot platform consisting of a collaborative robot and projector-camera system. Diesel engine parts on the table are used in the experiments and given by a local automotive factory.

5.3.2 AR-based UI

A graphical UI was created for intuitive interaction between the human and robot and it contains the following interaction components:

- **Danger zone:** During normal operation the danger zone is always visible and colored as a solid red boundary isolating the robot. The main target of the visualized safety zone is to increase the human awareness and confidence by indicating which regions the human operator should avoid during the task. In addition, changed regions due to human operations on the workspace are highlighted by yellow markers.
- **UI buttons:** The implemented UI contains the following interaction buttons 1) *GO* and *STOP* buttons to start and stop the robot; 4) *CONFIRM* button

to verify and add changed regions to the current model; 5) *ENABLE* button that needs to be pressed simultaneously with the *GO* and *CONFIRM* buttons to take effect. The enable button was created to guarantee that the human operator does not accidentally start or confirm objects by for instance leaning over the table where the buttons are projected.

- **Information menu:** Graphical bars and boxes are used to display task related information and instructions during the task. For instance, the robot planned operations are illustrated using text and image.

The graphical UI was implemented to two different hardware: projector-camera and HoloLens. The UI components and layout were the same for both hardware. For the projector-camera, the interface was a projected color display, containing all the components as 2D objects. Buttons were positioned in the vicinity of the operator and visualized as a colored circles including textual identifiers. Information menus were rectangular objects with visual and textual information to inform the human operator. The robot and camera are calibrated through common pattern-based procedures. Projector image plane and the robot xy-plane are related using a global homography matrix $\boldsymbol{H}$. Then homogeneous coordinate $\vec{x} = (x, y, 1)$ of a pixel on the generated UI image is projected to the display surface (e.g. robot xy-plane) on the scene $\vec{x'} = (x', y', 1)$ by the formula

$$w \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \boldsymbol{H}\vec{x} = \begin{bmatrix} h_0 & h_1 & h_2 \\ h_3 & h_4 & h_5 \\ h_6 & h_7 & h_8 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad , \tag{5.11}$$

where w is the scaling factor.

In HoloLens, the interaction buttons are displayed as semi-transparent spheres that are positioned similar to the projector-camera UI. The instructions and robot related information is displayed on a floating 2D plane that is positioned to the side of the table. The safety region is rendered as a solid polygonal mesh having semi-transparent red texture. Using the 2D coordinates of the safety boundary and a fixed fence height, the fence mesh is constructed from rectangular quadrilaterals that are further divided to two triangles for the HoloLens rendering software. The UI component and the virtual fence coordinates $\vec{p} = (x, y, z)$ are defined in the robot

frame and transformed to the HoloLens frame by

$$\vec{p}' = T_{AR}^{R} T_{H}^{AR} \vec{p} \ , \tag{5.12}$$

where T_{AR}^{R} is a known static transformation between the robot and an AR marker (set manually to the workspace) and T_{H}^{AR} is the transformation between the marker and the user holographic frame. Once the pose has been initialized the marker can be removed and during run time T_{H}^{AR} is updated by HoloLens software. The data exchange between HoloLens and PC is done using wireless TCP/IP. We implemented a Linux server that synchronizes data from the safety model to UI and back.

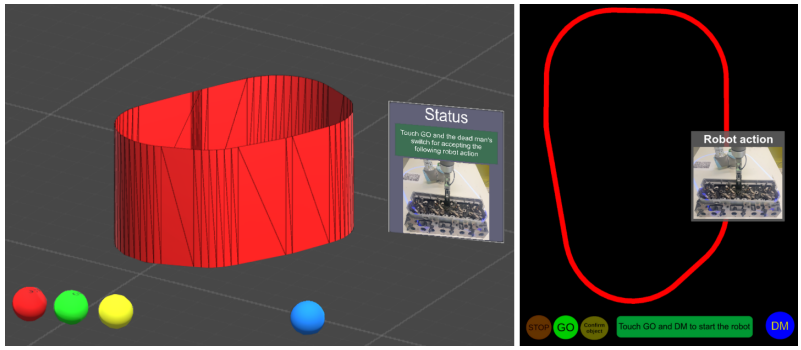


Figure 5.3 Graphical UIs: HoloLens setup rendered in Unity3D engine (left) and projector-mirror as a 2D color image (right).

5.4 Experiments

5.4.1 Task

Two different tasks were created to benchmark the safety model and UIs: *baseline assembly* and *diesel engine assembly*.

Baseline assembly. The baseline task is adopted from the well-known *Cranfield benchmark* [26] from which we selected 7 parts and defined which assembly stages are made by a robot (R) and which by a human(H). See the whole task allocation in Fig. 5.4. The sub-tasks in the assembly are the following: Task 1) The robot brings

the back plate to the shared workspace and goes back to collect the front plate, Task 2) the human co-worker inserting the bolts into the back plate and Task 3) the robot brings and installs the front plate and the task is finished. All the tasks are dependent i.e. the previous task has to be done before starting the next one.

Diesel engine assembly. The industry relevant task is adopted from a local diesel engine manufacturer which is previously done solely by humans. The task is particularly interesting as one of the sub-tasks is to install a rocker shaft that weights 4.3 kg and would therefore benefit from HRC. The task is illustrated in Figure 5.5 which also shows the five dependent sub-tasks and the task allocation (H denotes the human operator and R the robot): Task 1) Install 8 rocker arms (H), Task 2) Install the motor frame (R), Task 3) Insert 4 frame screws (H), Task 4) Install the rocker shaft (R+H) and Task 5) Insert the nuts on the shaft (H). Task 4 is collaborative in the sense that the robot brings the rocker shaft and activates force mode allowing physical hand-guidance of the end effector. In the force mode, the robot applies just enough force to overcome the gravitational force of the object while still allowing the human to guide the robot arm for accurate positioning.

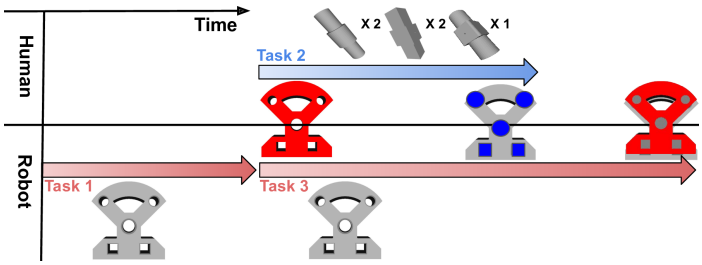


Figure 5.4 The Cranfield assembly task used in the experiments. The human task (blue) is to insert five bolts on the object while robot (red) is responsible for bringing and installing the side parts of the object.

5.4.2 Methods

For the experiments four different setups were implemented: *HoloLens*, *projector-camera*, *projector-baseline* and *non-collaborative baseline*. HoloLens and projector-camera are based on the proposed safety model for collaborative manufacturing. Both methods use a single depth sensor to monitor and update the HRC zones de-

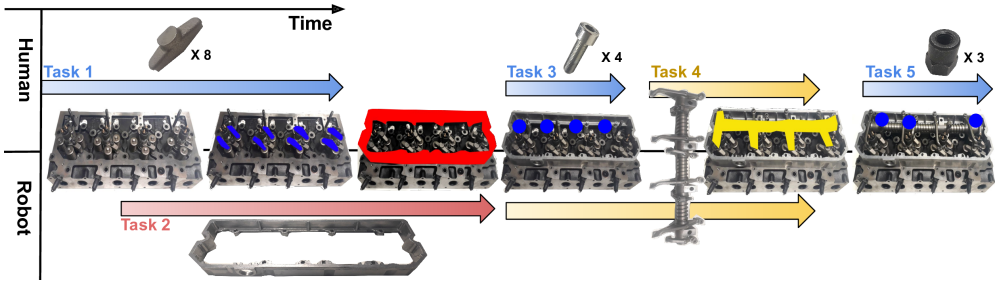


Figure 5.5 The engine assembly task consists of five sub-tasks (Task 1-5) that are conducted by the operator (blue) or the robot (red) or both (yellow). Task 4 is the collaborative sub-task where a rocker shaft is held by the robot and carefully positioned by the operator.

fined on the workspace model. The communication between the human and robot is realized using the proposed UI (Section 5.3.2), implemented on the two different hardware.

The projector-baseline is similar to [138, 139] and uses RGB and projector to detect safety violations. The method does not specifically model or update the workspace but assumes it is a planar table. The danger zone boundaries are projected to the workspace and safety violations are detected by comparing the projected boundary to a simulated boundary. The method is based on two different masks: current-state mask (measured by the RGB camera) and expected-state mask (simulated using robot movement and known workspace structure). The boundary was simplified by using a single static line (see video[1]) without compromising the safety performance. UI is similar to projector-camera.

The non-collaborative baseline is based on the current practices in manufacturing – the human and robot cannot operate in the same work space simultaneously. In the setting, the operator must stay 4 m apart from the robot when the robot is moving and the operator is allowed to enter the workspace only when the robot is not moving. Safety in the non-collaborative baseline is ensured by an enabling switch button which the operator needs to press all the time for the robot to be operational. The baseline does not contain any UI components, but the users are provided with textual descriptions for all sub-tasks.

[1]https://youtu.be/CFKKANvWc3A

5.4.3 Performance metrics

The data collection from experiments included recordings of performance times for quantitative evaluation and filled questionnaires to asses the qualitative aspects during the tasks. During the experiments, the cooperation zone Z_c was extended to cover the whole shared workspace S and the robot speed was approximately 50% of the full speed at all time.

Quantitative performance. For task performance evaluation we selected two different metrics: *total execution time* and *robot idle time*. The total execution time measures how long it takes for the human and robot to finish the whole task. The idle time measures how long during the task the robot (including gripper) was doing nothing.

Qualitative performance. In order to evaluate physical and mental stress aspects of the human co-workers during the tasks, a questionnaire was created including 13 different questions (see more details in [**P5**]). The questions were selected to cover safety, ergonomics and mental stress experience as defined in Salvendy et al. [116] and autonomy, competence, and relatedness in Deci et al. [29]. Users were asked to score each question using the scale from 1 (totally disagree) to 5 (totally agree). No personal data was collected during the experiment.

5.5 Results

Quantitative performance. Quantitative performance was measured in both tasks and the results are shown in Fig. 5.6. In the terms of total execution and robot idle time, the proposed safety model outperformed all the other methods. On cranfield dataset, the projector-camera was compared against the baseline safety system and the total improvement was 12.4% in overall performance and 40% in robot idle time. The main reason for the improvements is the fact that our safety model allows parallel working in the same shared workspace where as the baseline does not.

The findings were verified on the diesel engine assembly task. This time the safety model was integrated also with HoloLens-based UI in addition to projection-based. The AR methods were compared against the non-collaborative baseline. Both AR-

based interactive systems outperformed the baseline where the robot was not moving in the same workspace with an operator. On average, the AR-based systems were 21 — 24% and 57 — 64% faster than the baseline in the terms of the total execution time and the robot idle time, respectively. The performance time difference between the two AR-based UIs was marginal.

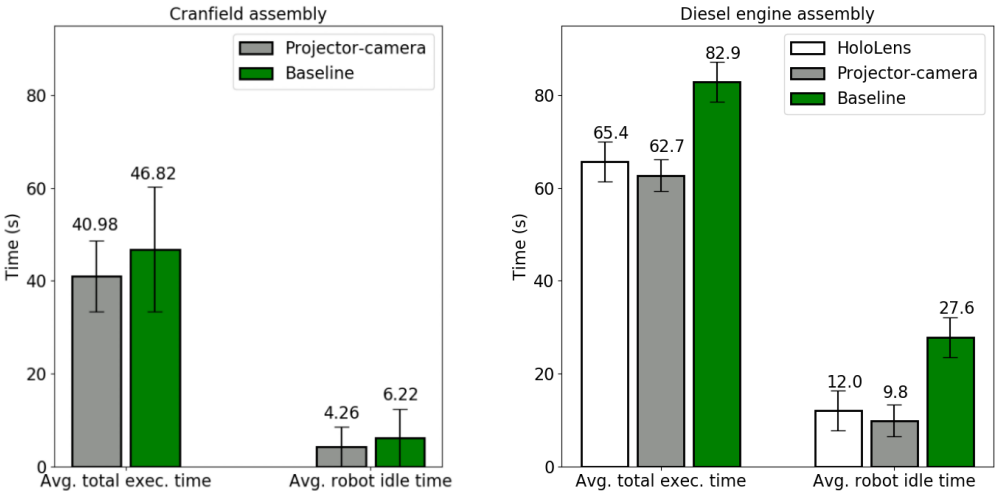


Figure 5.6 Average total task execution and robot idle times from two different tasks: Cranfield (left) and diesel engine (right) assembly.

Qualitative performance. The subjective evaluations were conducted with 20 inexperienced volunteer university students and all the findings are presented in Table 5.7. The evaluation was conducted in the diesel assembly task and the two AR-based methods and the non-collaborative baseline were included in the comparison. The overall impression is that the projector-based system outperforms the two others (HoloLens and non-collaborative baseline), but surprisingly HoloLens is found inferior to the baseline in many safety related questions. The projector-based method is considered the safest and the HoloLens-based method the most unsafe with a clear margin.

Ergonomics-wise HoloLens and projector-camera were superior likely to the fact that they provided help in installing the heavy rocker shaft. The autonomy numbers are similar for all methods, but the projector-based is found the easiest to work with. The users also found the HoloLens and projector-based methods in the terms of com-

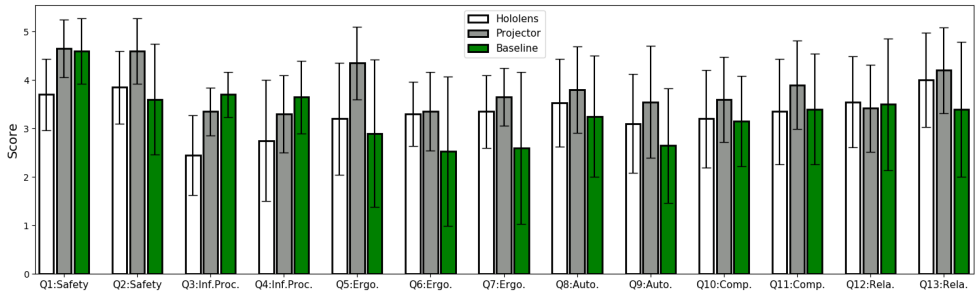


Figure 5.7 Average scores for the questions Q1-Q13 used in the user studies [**P5**]. Higher score means better performance and scores for the questions Q3, Q4, Q6, Q7 and Q10 are inverted for better readability.

petence and relatedness more suitable than the baseline. Overall, the projector-based AR interaction in collaborative manufacturing was found safer and more ergonomic than the baseline without AR interaction and also the HoloLens-based AR.

5.6 Summary

In this chapter, a computation model of the shared workspace in HRC manufacturing was introduced. The model defines and monitors four spatial zones in the workspace, providing safe and efficient interaction between the robot and human. Moreover, the chapter described an UI for HRC in industrial manufacturing that was implemented on two different hardware for AR, a projector-camera and wearable AR gear (HoloLens).

The model and UIs were experimentally evaluated in two different assembly tasks and results from quantitative and qualitative evaluations with respect to performance, safety and ergonomics, and against two baseline methods were reported. In both tasks, the AR-based systems were found superior in performance to the baselines without a shared workspace. However, the users found the projector-camera system clearly more plausible for manufacturing work than the HoloLens setup. The other AR research studies considering traditionally conveyed AR e.g. via monitors or tablets reported that AR technologies receives positive feedback from the potential users. The studies agree with this indication, except when using wearable AR such as HMDs. The wearable AR requires still more technical maturity (in de-

sign, safety and software side) in order to be considered suitable for industrial environments.

6 CONCLUSION

In this thesis, a number of contributions related to computer vision and robotics have been presented. The thesis was divided into three major categories: 1) object class matching, 2) 6D object pose estimation and 3) human-robot collaboration. The handcrafted 2D local features, e.g. SIFT feature, have been the basis for many vision-based applications, where robust image matches between images of the same scene from various viewpoints have to be established. In Chapter 2, the first contribution was to extend the well-known local feature benchmark from wide baseline matching to object class matching. In particular, we were interested in how well the recent methods find matches between objects from the same base class but dissimilar appearance. The overall impression was that the detectors performed well but the descriptors ability to describe semantically meaningful parts similarly between two objects was poor. Based on the results, the detector meta-parameters have high impact on the performance and specialized descriptors for visual class parts and regions are needed.

Two of the contributions were related to 6D pose estimation and especially in their applications in robotics. In Chapter 3, a complete description of a 3D-to-3D correspondence based pose estimation pipeline was given. The pipeline relies on local feature representation of object inputs, feature matching and geometric verification of matched features. Based on the verified matches, the object model can be localized from the sensor measurement. One of the disadvantages of the pipeline is that the estimation performance relies heavily on the object surface geometry. This is problematic as many real life objects share similar appearance or have simple surface structure, leading incorrectly established point pair matches. To address these problems, two different algorithms for exploiting the object surface and removing unreliable points were proposed. In the experiments the relatively simple algorithms were able to select a robust sub-set of matches against estimation failures and improve the overall pipeline accuracy. The experiments were repeated on a

much bigger dataset and verified the following findings from the earlier experiments: 1) among the pose estimation methods Geometrical Consistency Grouping provides the overall best performance and 2) meta-parameters of each method are by default far away from the optimal ones. However, the experiments revealed that the robustifying methods do not systematically improve the results and can sometimes lead to clearly inferior results.

At the end of Chapter 3, we revisited the evaluation of vision based object pose estimation methods for robotics. We criticized that the existing evaluation metrics measure the "goodness" of a pose estimate solely based on the spatial alignment of two geometric objects which does not directly indicate the estimate performance on real robotic task. Therefore, we proposed a completely new metric based on a statistical formulation of the task success probability given an estimated object pose. In the experiments, we quantitatively demonstrated the proposed metric to be a more reasonable metric for an industrial assembly task than the popular error metric. In addition, we proposed an industry relevant dataset containing hundreds of test images with ground truth annotations. As a summary, the novel metric and dataset provide basis for more realistic evaluation of object pose estimation methods without requiring a physical setup. In the future work, we will continue to promote more practical research on robotics and 3D object pose estimation by including other types of assembly tasks into the benchmark. In addition, the future work includes investigating the metric on other robotic domains, such as navigation, i.e. what is the error tolerance in localization for successful maneuvering in narrow spaces.

Another important contributions, related to human-robot collaboration, were presented in the final chapters of the thesis. In human-robot collaboration the human co-worker operates in fenceless and shared environment next to the robot. In such scenario, novel safety approaches are needed for collision detection while still allowing close collaboration. During the course of work, a complete HRC safety model for collaborative manufacturing was presented. The model is based on several dynamic HRC zones, each having own safety properties. The safety model was experimentally evaluated and the results verified the potential of HRC be more efficient alternative compared to current practices in manufacturing. Finally, the usefulness and readiness level of AR-based techniques, image projector and head-mounted display (HoloLens), as an UI medium in manufacturing task was evaluated. Based on the subjective evaluations, the projector was found more suitable for supporting and

instructing the human operator compared to the other methods and surprisingly, most of the participants considered HoloLens to be unsafe and cumbersome. However, Microsoft has recently released the latest generation of HMD (HoloLens 2) that has improved on the previous technical, visual, and functional aspects of HoloLens 1. It is an open question whether or not the new device improves the user experience in manufacturing tasks.

To conclude, the field of robotics is challenging branch of engineering and science that combines multiple research fields including mechanical engineering, electrical engineering, and computer science, to name of few. In this thesis, several problems related to vision-aided robotics were addressed and solutions for the problems were proposed. Many of the works were demonstrated in realistic tasks and we believe that our work can be used as a starting point for new systems for many practical problems.

REFERENCES

[1] A. Agarwal and B. Triggs. Hyperfeatures – Multilevel Local Coding for Visual Recognition. *European Conference on Computer Vision*. Springer. 2006, 30–43.

[2] A. Alahi, R. Ortiz and P. Vandergheynst. Freak: Fast retina keypoint. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, 510–517.

[3] R. S. Andersen, O. Madsen, T. B. Moeslund and H. B. Amor. Projecting robot intentions into human environments. *International Symposium on Robot and Human Interactive Communication*. IEEE. 2016, 294–301.

[4] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, 2911–2918.

[5] R. C. Arkin and T. R. Collins. *Skills impact study for tactical mobile robot operational units*. Tech. rep. Georgia Institute of Technology, 2002.

[6] F. Attneave. Some informational aspects of visual perception. *Psychol Rev* 61.3 (1954), 183–193.

[7] M. Bdiwi. Integrated sensors system for human safety during cooperating with industrial robots for handing-over and assembling tasks. *CIRP Annals* 23 (2014), 65–70.

[8] M. Bdiwi, M. Pfeifer and A. Sterzing. A new strategy for ensuring human safety during various levels of interaction with industrial robots. *CIRP Annals* 66 (2017), 453–456.

[9] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review* 94 (1987), 115–147.

[10] T. Birdal and S. Ilic. Point pair features based object detection and pose estimation revisited. *International Conference on 3D Vision*. IEEE. 2015, 527–535.

[11] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton and C. Rother. Learning 6d object pose estimation using 3d object coordinates. *European Conference on Computer Vision*. Springer. 2014, 536–551.

[12] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2016, 3364–3372.

[13] A. Buch, Y. Yang, N. Krüger and H. Petersen. In Search of Inliers: 3D Correspondence by Local and Global Voting. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2014, 2067–2074.

[14] A. G. Buch, L. Kiforenko and D. Kraft. Rotational subgroup voting and pose clustering for robust 3D object recognition. *International Conference on Computer Vision*. IEEE. 2017, 4137–4145.

[15] A. G. Buch, D. Kraft, J.-K. Kamarainen, H. G. Petersen and N. Krüger. Pose estimation using local structure-specific shape and appearance context. *International Conference on Robotics and Automation*. IEEE. 2013, 2080–2087.

[16] C. Byner, B. Matthias and H. Ding. Dynamic speed and separation monitoring for collaborative robot applications – Concepts and performance. *Robotics and Computer-Integrated Manufacturing* 58 (2019), 239–252.

[17] M. Calonder, V. Lepetit, C. Strecha and P. Fua. Brief: Binary robust independent elementary features. *European Conference on Computer Vision*. Springer. 2010, 778–792.

[18] Z. Cao, Y. Sheikh and N. K. Banerjee. Real-time scalable 6DOF pose estimation for textureless objects. *International Conference on Robotics and Automation*. IEEE. 2016, 2441–2448.

[19] M. Cefalo, E. Magrini and G. Oriolo. Parallel collision check for sensor based real-time motion planning. *International Conference on Robotics and Automation*. IEEE. 2017, 1936–1943.

[20] K. Chatfield, V. Lempitsky, A. Vedaldi and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. *BMVA*. 2011, 76.1–76.12.

[21] H. Chen and B. Bhanu. 3D free-form object recognition in range images using local surface patches. *Pattern Recognition Letters* 28.10 (2007), 1252–1262.

[22] J.-H. Chen and K.-T. Song. Collision-Free Motion Planning for Human-Robot Collaborative Safety under Cartesian Constraint. *International Conference on Robotics and Automation*. IEEE. 2018, 1–7.

[23] Y. Chen and G. G. Medioni. Object modeling by registration of multiple range images. *Image Vision Computing* 10.3 (1992), 145–155.

[24] Z. Chen, S. Czarnuch, A. Smith and M. Shehata. Performance evaluation of 3D keypoints and descriptors. *International Symposium on Visual Computing*. Springer. 2016, 410–420.

[25] C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu and S. Ramalingam. Voting-based pose estimation for robotic assembly using a 3D sensor. *International Conference on Robotics and Automation*. IEEE. 2012, 1724–1731.

[26] K. Collins, A. Palmer and K. Rathmill. Robot Technology and Applications. Ed. by K. Rathmill, P. MacConaill, P. O'Leary and J. Browne. 1985. Chap. The Development of a European Benchmark for the Comparison of Assembly Robot Programming Systems.

[27] C. M. Cyr and B. B. Kimia. A similarity-based aspect-graph approach to 3D object recognition. *International Journal of Computer Vision* 57.1 (2004), 5–22.

[28] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Conference on Computer Vision and Pattern Recognition*. Vol. 1. IEEE. 2005, 886–893.

[29] E. L. Deci, R. J. Vallerand, L. G. Pelletier and R. M. Ryan. Motivation and education: The self-determination perspective. *Educational psychologist* 26.3-4 (1991), 325–346.

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, 248–255.

[31] A. Doumanoglou, R. Kouskouridas, S. Malassiotis and T.-K. Kim. Recovering 6D object pose and predicting next-best-view in the crowd. *Conference on Computer Vision and Pattern Recognition*. 2016, 3583–3592.

[32] B. Drost, M. Ulrich, N. Navab and S. Ilic. Model globally, match locally: Efficient and robust 3D object recognition. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, 998–1005.

[33] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii and T. Sattler. D2-net: A trainable cnn for joint description and detection of local features. *Computer Vision and Pattern Recognition*. 2019, 8092–8101.

[34] J. Elsdon and Y. Demiris. Augmented Reality for Feedback in a Shared Control Spraying Task. *International Conference on Robotics and Automation*. IEEE. 2018, 1939–1946.

[35] M. Everingham, L. Gool, C. K. Williams, J. Winn and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision* 88.2 (2010), 303–338. ISSN: 0920-5691. DOI: 10.1007/s11263-009-0275-4. URL: http://dx.doi.org/10.1007/s11263-009-0275-4.

[36] F. Fabrizio and A. De Luca. Real-time computation of distance to dynamic obstacles with multiple depth sensors. *Robotics and Automation Letters* 2.1 (2016), 56–63.

[37] L. Fei-Fei, R. Fergus and P. Perona. One-shot learning of object categories. *Transactions on pattern analysis and machine intelligence* 28.4 (2006), 594–611.

[38] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24.6 (1981), 381–395.

[39] F. Flacco, T. Kröger, A. De Luca and O. Khatib. A depth space approach to human-robot collision avoidance. *International Conference on Robotics and Automation*. IEEE. 2012, 338–345.

[40] A. Flint, A. Dick and A. Van Den Hengel. Thrift: Local 3d structure recognition. *Digital Image Computing Techniques and Applications*. IEEE. 2007, 182–188.

[41] P. J. Flynn and A. K. Jain. CAD-based computer vision: from CAD models to relational graphs. *International Conference on Systems, Man and Cybernetics*. IEEE. 1989, 162–167.

[42] J. H. Friedman, J. L. Bentley and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software* 3.3 (1977), 209–226.

[43] R. K. Ganesan, Y. K. Rathore, H. M. Ross and H. B. Amor. Better teaming through visual cues: How projecting imagery in a workspace can improve human-robot collaboration. *Robotics and Automation Magazine* 25.2 (2018), 59–71.

[44] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas and M. J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47.6 (2014), 2280–2292.

[45] J. de Gea Fernández, D. Mronga, M. Günther, T. Knobloch, M. Wirkus, M. Schröer, M. Trampler, S. Stiene, E. Kirchner, V. Bargsten et al. Multimodal sensor-based whole-body control for human-robot collaboration in industrial settings. *Robotics and Autonomous Systems* 94 (2017), 102–119.

[46] C. Gkournelos, P. Karagiannis, N. Kousi, G. Michalos, S. Koukas and S. Makris. Application of wearable devices for supporting operators in human-robot cooperative assembly tasks. *CIRP Annals* 76 (2018), 177–182.

[47] R. L. Graham and F. F. Yao. Finding the convex hull of a simple polygon. *Journal of Algorithms* 4.4 (1983), 324–331.

[48] S. A. Green, M. Billinghurst, X. Chen and J. G. Chase. Human-robot collaboration: A literature review and augmented reality approach in design. *International Journal of Advanced Robotic Systems* 5.1 (2008), 1.

[49] M. Gualtieri, A. Ten Pas, K. Saenko and R. Platt. High precision grasp pose detection in dense clutter. *International Conference on Intelligent Robots and Systems*. IEEE. 2016, 598–605.

[50] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan and N. M. Kwok. A comprehensive performance evaluation of 3D local feature descriptors. *International Journal of Computer Vision* 116.1 (2016), 66–89.

[51] R.-J. Halme, M. Lanz, J. Kämäräinen, R. Pieters, J. Latokartano and A. Hietanen. Review of vision-based safety systems for human-robot collaboration. *CIRP Annals* 72 (2018), 111–116.

[52] X.-F. Hana, J. S. Jin, J. Xie, M.-J. Wang and W. Jiang. A comprehensive review of 3d point cloud descriptors. *arXiv preprint arXiv:1802.02297* (2018).

[53] R. Hänsch, T. Weber and O. Hellwich. Comparison of 3D interest point detectors and descriptors for point cloud fusion. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2.3 (2014), 57.

[54] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[55] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua and N. Navab. Dominant orientation templates for real-time detection of texture-less objects. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, 2257–2264.

[56] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. *Asian Conference on Computer Vision*. Springer. 2012, 548–562.

[57] S. Hinterstoisser, V. Lepetit, N. Rajkumar and K. Konolige. Going further with point pair features. *European Conference on Computer Vision*. Springer. 2016, 834–848.

[58] G. Hirzinger, J. Butterfass, M. Fischer, M. Grebenstein, M. Hahnle, H. Liu, I. Schaefer and N. Sporer. A mechatronics approach to the design of light-weight arms and multifingered hands. *International Conference on Robotics and Automation*. IEEE. 2000, 46–54.

[59] T. Hodaň, J. Matas and Š. Obdržálek. On Evaluation of 6D Object Pose Estimation. *European Conference on Computer Vision* (2016), 606–619.

[60] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis et al. BOP: benchmark for 6D object pose estimation. *European Conference on Computer Vision*. Springer. 2018, 19–34.

[61] D. Q. Huy, I. Vietcheslav and G. S. G. Lee. See-through and spatial augmented reality – a novel framework for human-robot interaction. *International Conference on Control, Automation and Robotics*. IEEE. 2017, 719–726.

[62] *ISO 10218-1 – Robots and Robotic Devices – Safety Requirements For Industrial Robots – Part 1: Robots*. International Organization for Standardization. 2011.

[63] *ISO 10218-2 – Robots and Robotic Devices – Safety Requirements For Industrial Robots – Part 2: Robot Systems And Integration*. International Organization for Standardization. 2011.

[64] *ISO 13857 – Safety of machinery – Safety distances to prevent hazard zones being reached by upper and lower limbs*. International Organization for Standardization. 2019.

[65] *ISO/TS 15066 – Robots and Robotic Devices – Collaborative Robots*. International Organization for Standardization. 2016.

[66] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *Transactions on pattern analysis and machine intelligence* 21.5 (1999), 433–449.

[67] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), 922–923.

[68] W. Kehl, F. Milletari, F. Tombari, S. Ilic and N. Navab. Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation. *European Conference on Computer Vision*. Springer. 2016, 205–220.

[69] W. Kehl, F. Tombari, N. Navab, S. Ilic and V. Lepetit. Hashmod: A hashing method for scalable 3D object detection. *arXiv preprint arXiv:1607.06062* (2016).

[70] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Autonomous Robot Vehicles*. Springer, 1986, 396–404.

[71] E. Kim, R. Kirschner, Y. Yamada and S. Okamoto. Estimating probability of human hand intrusion for speed and separation monitoring using interference theory. *Robotics and Computer-Integrated Manufacturing* 61 (2020), 101819.

[72] T. Kinnunen, J.-K. Kamarainen, L. Lensu, J. Lankinen and H. Kalviainen. Making visual object categorization more challenging: Randomized caltech-101 data set. *International Conference on Pattern Recognition*. IEEE. 2010, 476–479.

[73] A. Krizhevsky, I. Sutskever and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*. 2012, 1097–1105.

[74] J. Lankinen, V. Kangas and J.-K. Kamarainen. A comparison of local feature detectors and descriptors for visual object categorization by intra-class repeatability and matching. *International Conference on Pattern Recognition*. 2012, 780–783.

[75] J. Lankinen and J.-K. Kämäräinen. Local Feature Based Unsupervised Alignment of Object Class Images. *British Machine Vision Conference*. Vol. 1. 2. 2011, 5.

[76] S. Lanser, O. Munkelt and C. Zierl. Robust video-based object recognition using CAD models. *Intelligent Autonomous Systems*. Citeseer. 1995, 529–536.

[77] P. A. Lasota, T. Fong, J. A. Shah et al. A survey of methods for safe human-robot interaction. *Foundations and Trends® in Robotics* 5.4 (2017), 261–349.

[78] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. *International Conference on Computer Vision*. IEEE. 2005, 1482–1489.

[79] S. Leutenegger, M. Chli and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. *International Conference on Computer Vision*. IEEE. 2011, 2548–2555.

[80] J. Li and N. M. Allinson. A comprehensive review of current local features for computer vision. *Neurocomputing* 71.10-12 (2008), 1771–1787.

[81] C. Liu and M. Tomizuka. Robot safe interaction system for intelligent industrial co-robots. *arXiv preprint arXiv:1808.03983* (2018).

[82] H. Liu, T. Fang, T. Zhou, Y. Wang and L. Wang. Deep learning-based multimodal control interface for human-robot collaboration. *CIRP Annals* 72 (2018), 3–8.

[83] H. Liu and L. Wang. An AR-based worker support system for human-robot collaboration. *Procedia Manufacturing* 11 (2017), 22–30.

[84] H. Liu and L. Wang. Human motion prediction for human-robot collaboration. *Journal of Manufacturing Systems* 44 (2017), 287–294.

[85] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Conference on Computer Vision*. IEEE. 2004, 91–110.

[86] E. Magrini, F. Ferraguti, A. J. Ronga, F. Pini, A. De Luca and F. Leali. Human-robot coexistence and interaction in open industrial cells. *Robotics and Computer-Integrated Manufacturing* 61 (2020), 101846.

[87] J. Mainprice and D. Berenson. Human-robot collaborative manipulation planning using early prediction of human motion. *International Conference on Intelligent Robots and Systems*. IEEE. 2013, 299–306.

[88] S. Makris, P. Tsarouchi, D. Surdilovic and J. Krüger. Intuitive dual arm robot programming for assembly operations. *CIRP Annals* 63 (2014), 13–16.

[89] E. Mariotti, E. Magrini and A. De Luca. Admittance Control for Human-Robot Interaction Using an Industrial Robot Equipped with a F/T Sensor. *International Conference on Robotics and Automation*. IEEE. 2019, 6130–6136.

[90] J. A. Marvel and R. Norcross. Implementing speed and separation monitoring in collaborative robot workcells. *Robotics and Computer-Integrated Manufacturing* 44 (2017), 144–155.

[91] E. Matheson, R. Minto, E. G. Zampieri, M. Faccio and G. Rosati. Human-Robot Collaboration in Manufacturing Applications: A Review. *Robotics* 8.4 (2019), 100.

[92] A. Mian, M. Bennamoun and R. Owens. Three-dimensional model-based object recognition and segmentation in cluttered scenes. *Transactions on pattern analysis and machine intelligence* 28.10 (2006).

[93] G. Michalos, N. Kousi, P. Karagiannis, C. Gkournelos, K. Dimoulas, S. Koukas, K. Mparis, A. Papavasileiou and S. Makris. Seamless human robot collaborative assembly – An automotive case study. *Mechatronics* 55 (2018), 194–211.

[94] G. Michalos, S. Makris, P. Tsarouchi, T. Guasch, D. Kontovrakis and G. Chryssolouris. Design considerations for safe human-robot collaborative workplaces. *CIRP Annals* 37 (2015), 248–253.

[95] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *Transactions on pattern analysis and machine intelligence* 27.10 (2005), 1615–1630.

[96] K. Mikolajczyk, B. Leibe and B. Schiele. Local features for object class recognition. *International Conference on Computer Vision*. IEEE. 2005, 1792–1799.

[97] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision* 65.1-2 (2005), 43–72.

[98] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *International Conference on Computer Vision Theory and Applications* 331-340 (2009), 2.

[99] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *Transactions on pattern analysis and machine intelligence* 36.11 (2014), 2227–2240.

[100] M. Muja, R. B. Rusu, G. Bradski and D. G. Lowe. Rein – A fast, robust, scalable recognition infrastructure. *International Conference on Robotics and Automation*. IEEE. 2011, 2939–2946.

[101] E. Nowak, F. Jurie and B. Triggs. Sampling strategies for bag-of-features image classification. *European Conference on Computer Vision*. Springer. 2006, 490–503.

[102] J. Papon, A. Abramov, M. Schoeler and F. Wörgötter. Voxel Cloud Connectivity Segmentation – Supervoxels for Point Clouds. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2013, 2027–2034.

[103] F. C. Park and B. J. Martin. Robot sensor calibration: solving AX=XB on the Euclidean group. *Transactions on Robotics* 10.5 (1994), 717–721.

[104] M. Pauly, M. Gross and L. P. Kobbelt. Efficient simplification of point-sampled surfaces. *Proceedings of the conference on Visualization*. IEEE Computer Society. 2002, 163–170.

[105] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *International Conference on Robotics and Automation*. IEEE. 2016, 3406–3413.

[106] C. Pohlt, F. Haubner, J. Lang, S. Rochholz, T. Schlegl and S. Wachsmuth. Effects on User Experience During Human-Robot Collaboration in Industrial Scenarios. *International Conference on Systems, Man, and Cybernetics*. IEEE. 2018, 837–842.

[107] M. P. Polverini, A. M. Zanchettin and P. Rocco. A computationally efficient safety assessment for collaborative robotics applications. *Robotics and Computer-Integrated Manufacturing* 46 (2017), 25–37.

[108] H. Rajnathsing and C. Li. A neural network based monitoring system for safety in shared workspace human-robot collaboration. *Industrial Robot: An International Journal* 45.4 (2018), 481–491.

[109] J. Redmon, S. Divvala, R. Girshick and A. Farhadi. You only look once: Unified, real-time object detection. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2016, 779–788.

[110] S. Robla-Gómez, V. M. Becerra, J. R. Llata, E. Gonzalez-Sarabia, C. Torre-Ferrero and J. Perez-Oria. Working together: A review on safe human-robot collaboration in industrial environments. *IEEE Access* 5 (2017), 26754–26773.

[111] E. Rosen, D. Whitney, E. Phillips, G. Chien, J. Tompkin, G. Konidaris and S. Tellex. Communicating and controlling robot arm motion intent through mixed-reality head-mounted displays. *The International Journal of Robotics Research* (2019), 0278364919842925.

[112] E. Rublee, V. Rabaud, K. Konolige and G. Bradski. ORB: An efficient alternative to SIFT or SURF. *International Conference on Computer Vision*. IEEE. 2011, 2564–2571.

[113] R. B. Rusu. Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz* 24.4 (2010), 345–348.

[114] R. B. Rusu, N. Blodow and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. *International Conference on Robotics and Automation*. IEEE. 2009, 3212–3217.

[115] R. B. Rusu, N. Blodow, Z. C. Marton and M. Beetz. Aligning point cloud views using persistent feature histograms. *International Conference on Intelligent Robots and Systems*. IEEE. 2008, 3384–3391.

[116] G. Salvendy. *Handbook of human factors and ergonomics*. John Wiley & Sons, 2012.

[117] S. Sato and S. Sakane. A human-robot interface using an interactive hand pointer that projects a mark in the real work space. *International Conference on Robotics and Automation*. Vol. 1. IEEE. 2000, 589–595.

[118] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic et al. Benchmarking 6dof outdoor visual localization in changing conditions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 8601–8610.

[119] A. Saxena, L. Wong, M. Quigley and A. Y. Ng. A vision-based system for grasping novel objects in cluttered environments. *Robotics research*. Springer, 2010, 337–348.

[120] S. Se, D. Lowe and J. Little. Global localization using distinctive visual features. *International Conference on Intelligent Robots and Systems*. IEEE. 2002, 226–231.

[121] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[122] I. Sipiran and B. Bustos. Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes. *The Visual Computer* 27.11 (2011), 963.

[123] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. *International Conference on Computer Vision*. IEEE. 2003, 1470.

[124] B. Steder, R. B. Rusu, K. Konolige and W. Burgard. NARF: 3D range image features for object recognition. *International Conference on Intelligent Robots and Systems*. Vol. 44. IEEE. 2010.

[125] A. Tejani, D. Tang, R. Kouskouridas and T.-K. Kim. Latent-class hough forests for 3D object detection and pose estimation. *European Conference on Computer Vision*. Springer. 2014, 462–477.

[126] B. Tekin, S. N. Sinha and P. Fua. Real-time seamless single shot 6d object pose prediction. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2018, 292–301.

[127] F. Tombari and L. Di Stefano. Object recognition in 3D scenes with occlusions and clutter by Hough voting. *Pacific-Rim Symposium on Image and Video Technology*. IEEE. 2010, 349–355.

[128] F. Tombari, S. Salti and L. Di Stefano. Unique signatures of histograms for local surface description. *European Conference on Computer Vision*. Springer. 2010, 356–369.

[129] F. Tombari, S. Salti and L. Di Stefano. A combined texture-shape descriptor for enhanced 3D feature matching. *International Conference on Image Processing*. IEEE. 2011, 809–812.

[130] P. Tsarouchi, S. Makris and G. Chryssolouris. Human–robot interaction review and challenges on task planning and programming. *International Journal of Computer Integrated Manufacturing* 29.8 (2016), 916–931.

[131] T. Tuytelaars, K. Mikolajczyk et al. Local invariant feature detectors: A survey. *Foundations and trends® in computer graphics and vision* 3.3 (2008), 177–280.

[132] T. Tuytelaars and C. Schmid. Vector Quantizing Feature Space with a Regular Lattice. *International Conference on Computer Vision*. IEEE. 2007, 1–8.

[133] T. Tuytelaars and L. Van Gool. Matching Widely Separated Views Based on Affine Invariant Regions. *International Journal of Computer Vision* 59.1 (2004), 61–85. ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000020671.28016.e8. URL: http://dx.doi.org/10.1023/B:VISI.0000020671.28016.e8.

[134] V. V. Unhelkar, P. A. Lasota, Q. Tyroller, R.-D. Buhai, L. Marceau, B. Deml and J. A. Shah. Human-aware robotic assistant for collaborative assembly: Integrating human motion prediction with planning in time. *Robotics and Automation Letters* 3.3 (2018), 2394–2401.

[135] F. Vicentini, M. Giussani and L. M. Tosatti. Trajectory-dependent safe distances in human-robot interaction. *Emerging Technology and Factory Automation*. IEEE. 2014, 1–4.

[136] J. Vidal, C.-Y. Lin and R. Martí. 6D pose estimation using an improved method based on point pair features. *International Conference on Control, Automation and Robotics*. IEEE. 2018, 405–409.

[137] U. Viereck, A. t. Pas, K. Saenko and R. Platt. Learning a visuomotor controller for real world robotic grasping using simulated depth images. *arXiv preprint arXiv:1706.04652* (2017).

[138] C. Vogel, M. Poggendorf, C. Walter and N. Elkmann. Towards safe physical human-robot collaboration: A projection-based safety system. *International Conference on Intelligent Robots and Systems*. IEEE. 2011, 3355–3360.

[139] C. Vogel, C. Walter and N. Elkmann. Safeguarding and supporting future human-robot cooperative manufacturing processes by a projection-and camera-based technology. *Procedia Manufacturing* 11 (2017), 39–46.

[140] L. Wang, R. Gao, J. Váncza, J. Krüger, X. V. Wang, S. Makris and G. Chryssolouris. Symbiotic human-robot collaborative assembly. *CIRP Annals* 68 (2019), 701–726.

[141] P. Wang, H. Liu, L. Wang and R. X. Gao. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. *CIRP Annals* 67 (2018), 17–20.

[142] X. V. Wang, A. Seira and L. Wang. Classification, personalised safety framework and strategy for human-robot collaboration. *International Conference on Computers and Industrial Engineering*. Curran Associates Inc. 2018.

[143] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3d pose estimation. *Conference on Computer Vision and Pattern Recognition*. IEEE. 2015, 3109–3118.

[144] J. Yang, K. Xian, P. Wang and Y. Zhang. A Performance Evaluation of Correspondence Grouping Methods for 3D Rigid Data Matching. *Transactions on pattern analysis and machine intelligence* (2019).

[145] Z. Zhang. A flexible new technique for camera calibration. *Transactions on pattern analysis and machine intelligence* 22 (2000).

[146] X. Zhao and J. Pan. Considering Human Behavior in Motion Planning for Smooth Human-Robot Collaboration in Close Proximity. *International Symposium on Robot and Human Interactive Communication*. IEEE. 2018, 985–990.

[147] Y. Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. *International Conference on Computer Vision*. IEEE. 2009, 689–696.

[148] H. Zhou, T. Sattler and D. W. Jacobs. Evaluating local features for day-night matching. *European Conference on Computer Vision*. Springer. 2016, 724–736.

PUBLICATIONS

PUBLICATION

I

A comparison of feature detectors and descriptors for object class matching

A. Hietanen, J. Lankinen, J.-K. Kämäräinen, A. G. Buch and N. Krüger

Neurocomputing 184.1 (2016), 3–12

A Comparison of Feature Detectors and Descriptors for Object Class Matching

Antti Hietanen, Jukka Lankinen, Joni-Kristian Kämäräinen[1]

Department of Signal Processing, Tampere University of Technology

Anders Glent Buch, Norbert Krüger

Maersk Mc-Kinney Moller Institute, University of Southern Denmark

Abstract

Solid protocols to benchmark local feature detectors and descriptors were introduced by Mikolajczyk et al. [1, 2]. The detectors and descriptors are popular tools in object class matching, but the wide baseline setting in the benchmarks does not correspond to class-level matching where appearance variation can be large. We extend the benchmarks to the class matching setting and evaluate state-of-the-art detectors and descriptors with Caltech and ImageNet classes. Our experiments provide important findings with regard to object class matching: 1) the original SIFT is still the best descriptor; 2) dense sampling outperforms interest point detectors with a clear margin; 3) detectors perform moderately well, but descriptors' performance collapse; 4) using multiple, even a few, best matches instead of the single best has significant effect on the performance; 5) object pose variation degrades dense sampling performance while the best detector (Hessian-affine) is unaffected. The performance of the best detector-descriptor pair is verified in the application of unsupervised visual class alignment where state-of-the-art results are achieved. The findings help to improve the existing detectors and descriptors for which the framework provides an automatic validation tool.

Keywords: local descriptor, local detector, interest point, SIFT, SURF,

[1]joni.kamarainen@tut.fi; +358 50 300 1851; P.O.Box 553, FI-33101 Tampere, Finland

1. Introduction

Image feature detectors and descriptors are the tools in computer vision problems where point or region correspondences between images are needed. Ideally, they should tolerate pose variation, illumination changes, motion blur and other typical scene changes and distortions. That is the case, for example, in wide baseline matching [3], robot localization [4] and panorama image stitching [5]. In these cases, the feature correspondences are needed to match several views of same scenes and the detector and descriptor evaluations by Mikolajczyk and Schmid 2005 [1] and Mikolajczyk et al. 2005 [2] help to find the most suitable detector-descriptor pair. A distinct application of feature-based matching is visual object classification and detection, where instances of object classes must be identified and localized in input images. In that case, the visual appearance variation can be very large as compared to fixed scenes, and thus, the original evaluations are not directly applicable.

Figure 1: Numbers of descriptor matches between two random class examples.

Various methods have been proposed for detecting interest points/regions and to construct descriptors from them, most of which are designed with a different application in mind. Recently, fast detectors and descriptors have been proposed: SURF [6], FREAK [7], ORB [8], BRISK [9], BRIEF [10] and LIOP [11]. In [1] detectors were evaluated by their repeatability ratios and total number of correspondences over several views of scenes and with various imaging distortion types. In [2] descriptors were evaluated by their matching rates for the same views. Comparisons on object classification were reported in [12] and [13], but they were tied to a single approach, visual Bag-of-Words (BoW). Our main contributions are:

- We introduce intuitive detector and descriptor evaluation frameworks by extending the detector and descriptor benchmarks in [1, 2] to intra-class repeatibility and matching.
- We evaluate the recent and popular detectors and descriptors and their various implementations with the proposed framework.
- We investigate the effect of using multiple best matches ($K = 1, 2, \ldots$) and introduce an alternative performance measure: *match coverage*.

From the experimental results on Caltech and ImageNet classes we arrive at the following important findings:

- Dense SIFT features are the best.
- Detectors generally perform well, but the ability of descriptors to match regions over visual class examples is poor (Fig. 1).
- Using multiple—even a few—best matches instead of the single best provides significant improvement.
- Dense grid sampling outperforms interest point detectors with a clear margin, but
- object pose variation can drastically affect dense sampling while the best detector (Hessian-affine) is unaffected.
- The original SIFT is still the best descriptor.

Source code for the evaluation framework will be published in the Web[2]. In addition, we verify our findings with the application of unsupervised object class alignment where the best detector-descriptor pair improves the state-of-the-art.

1.1. Related work

We believe that the general evaluation principles in [1, 2] also hold in the context of visual object classes: 1) *detectors which return the same object regions for class examples are good detectors* – detection repeatability; 2) *descriptors which match the same object regions between class examples are good descriptors* – match count/ratio. We refer to these repeating and matching regions as "category-specific landmarks". A qualitative measure to visualize descriptors ("HOGgles") was recently proposed by Vondrick et al. [14], but its main use is in visualization. More quantitative evaluations were reported by Zhang et al. [12] and Mikolajczyk et al. [13], but these were tied to a single methodology, the visual Bag-of-Words (BoW) [15, 16]. In this work, we show that the original evaluation principles can be adopted to obtain similar quantitative performance measures in general, comparable and intuitive forms to the original works of Mikolajczyk et al., and not tied to any specific approach.

2. Comparing Detectors

A good feature detector should detect local points or regions at the same locations of class examples to make it possible to match corresponding "parts". This criterion differs from [1], where detectors were evaluated over views of same scenes corresponding to specific object matching. In part-based object classification (e.g., [17]), the descriptors (parts) should match despite substantial variance in their visual appearance.

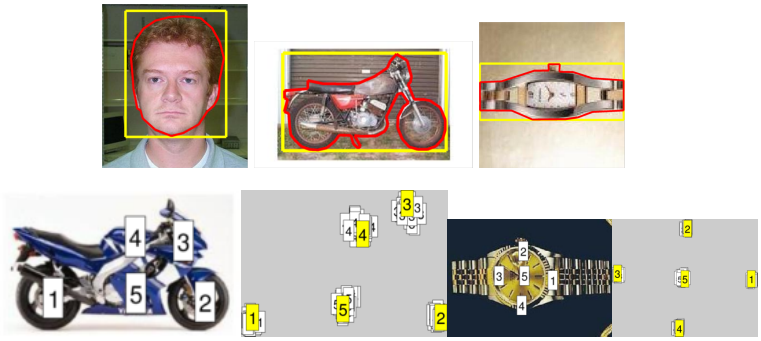


Figure 2: Top: example images with the provided ground truth (bounding boxes and foreground regions). Bottom: landmark examples and multiple landmarks projected onto a single image (the yellow tags).

2.1. Data

The experiments were conducted with the Caltech-101 [18] images. Caltech-101 is preferred as the baseline since objects' poses are roughly fixed that allows us to measure the effect of appearance variation without geometric pose noise. In the additional experiments we verify our results with randomly rotated versions of the Caltech images and the recent ImageNet database [19]. The foreground masks were used to remove features detected in the background (Fig. 2). Affine correspondence between category examples were established by manually annotating 5-12 landmarks per category and estimating the pair-wise image transformations using the direct linear transform [20] and linear interpolation. 25 random pairs from each class were repeatedly picked.

2.2. Feature detectors

The detectors for the experiments were selected among the best performing from our preliminary study [21] and the recently proposed detectors: BRIEF [10], BRISK [9], ORB [8] and FREAK [7]. The preliminary detectors were

[2] `https://bitbucket.org/kamarain/descriptor_vocbenchmark`

1. Two implementations of the difference of Gaussian: *sift* and *dog-vireo*
2. Harris-Laplace: *harlap-vireo*
3. Laplacian of Gaussian (log): *log-vireo*
4. Three implementations of the Hessian-affine: *hessaff*, *hessaff-alt* and *hesslap-vireo*
5. Speeded-up robust features: *surf*
6. Maximally stable extremal regions: *mser*

The detectors are publicly available: **-vireo* implementations in Zhao's Lip-vireo toolkit (`http://code.google.com/p/lip-vireo`), *hessaff* and *hessaff-alt* (by Mikolajczyk) at `http://featurespace.org`, *surf* at the authors' [6] web site and *mser* and *sift* in the VLFeat toolbox (`http://vlfeat.org`). The best average repeatability was 33.7% for *dog-vireo* and the best number of corresponding regions 57.4 for *hesslap-vireo*. The best three detectors based on the both repeatability and number of regions were *hesslap-vireo* (30.6%, 57.4), *hessaff* (25.3%, 47.8) and log-vireo (26.3%, 46.5). We report results for the best: the hessaff detector.

The best result from the recent detectors was obtained with the ORB OpenCV implementation (`http://opencv.org`) which is included (*orb*). Moreover, dense sampling has replaced detectors in the top methods (Pascal VOC 2011 [22]) and we added the dense SIFT in VLFeat (`http://vlfeat.org`) to our evaluation (*dense*).

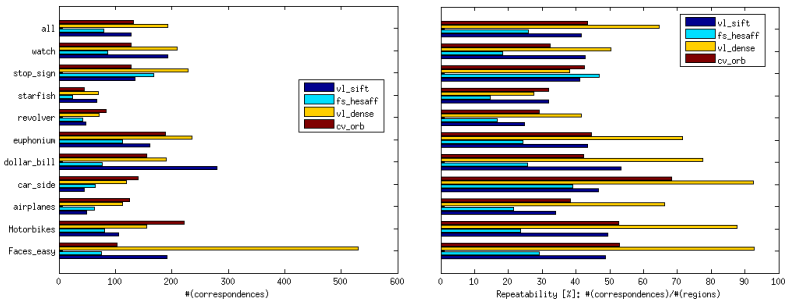
2.3. Performance measures and evaluation

For the detector performance evaluation, we adopted the procedure in [1] with the exception that interest points detected outside the object area (Fig. 2) are removed. For each image pair, points from the first image are projected onto the second image by the affine transformation estimated using the annotated landmarks. The interest points (regions) are described by 2D ellipses and when a transformed ellipse overlaps with an ellipse in the second image a correct correspondence is recorded. The number and rate of correspondences for each detector is of interest. A detector performs well if the total number is large and

has high precision if the ratio of correct matches is high. We used the parameter settings from [1]: 60% overlap threshold and normalization of the ellipses to the radius of 30 pixels. The normalization is required since the overlap area depends on the size of the ellipses.

The reported performance numbers are the average number of correspences between image pairs and the repeatability rate, i.e. the number of correspondences divided by the total number of points.

2.4. Results



(a) (b)

Detector	*Avg # of corr.*	*Avg. rep. rate*
vl_sift	127.5	41.6%
fs_hessaff	79.3	26.0%
cv_orb	132.0	43.5%
vl_dense	192.3	64.6%

(c)

Figure 3: Detector evaluation in object class matching. Meta-parameters were set to return on average 300 regions. (a) average number of corresponding regions, (b) repeatability rates, and (c) the overall results table.

It is noteworthy that this experiment differs from our preliminary work in the sense that instead of using the default parameters for each detector we

adjusted their meta-parameters to return on average 300 regions for each image (see Sec. 2.5 for further analysis). The results of the detector experiment are shown in Fig. 3. With the adjusted meta-parameters the difference between the detectors is less significant than in our preliminary work [21] (fs_hessaff 47.8/25.3%, vl_sift 16.2/21.5%) and the previous winner, Hessian-affine, is now the weakest. The dense sampling is clearly better than others, but otherwise the ORB detector seems attempting due to its speed. It is also noteworthy that without the parameter adjustment the results of the original SIFT detector would be by order of magnitude worse. Some less favorable properties of dense sampling are discussed in Sec. 4.4.

2.5. Detecting more regions

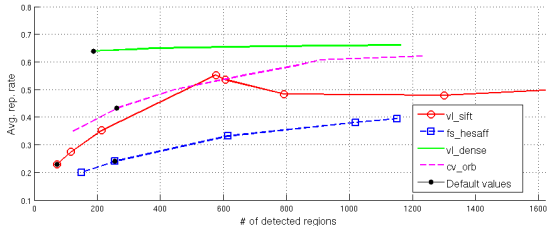


Figure 4: Detector repeatability as the function of the number of detected regions adjusted by the meta-parameters (defaults marked by black dots).

In the previous example, we adjusted detector meta-parameters to return on average 300 regions for each image. That made detectors produce very similar results while using the default parameters in our previous work lead to completely different interpretation. It is interesting to study whether we can exploit meta-parameters further to increase the number of corresponding regions. For ORB we adjusted the edge threshold, for Hessian-affine the feature density and the Hessian threshold, for SIFT the number of levels per octave, and for the dense the grid step size. We computed the detector repeatability rates as the functions of the number of detected regions (see Figure 4). As expected the

meta-parameters have almost no effect to the dense detection while Hessian-affine, ORB and especially SIFT clearly improve as the number of regions increase (SIFT regions saturate to the same locations approx. at 600 detected regions). For the most difficult classes in Fig. 3 (starfish and revolver) more regions is beneficial opening a novel research direction whether the detector parameters should be optimized for class detection?

3. Comparing Descriptors

A good region descriptor for object matching should be discriminative to match only correct regions, and also tolerate small appearance variation between the examples. The descriptor performances were obtained in the original work [2] by computing statistics of the correct and false matches. Between different class examples, descriptor matches are expected to be weaker due to increased appearance variation. For example, scooters and road bikes are both in the Caltech-101 motorbikes category, but their pair-wise similarity is much weaker than between two scooters or two road bikes.

3.1. Available descriptors

This experiment is conducted using detector-descriptor pairs. Our preliminary set of descriptors was:

1. Hessian-affine and SIFT
2. Hessian-affine and steerable filters
3. Vireo implementation of Hessian-affine and SIFT
4. Original SIFT detector and SIFT descriptor
5. Alternative (Vireo) implementation of SIFT and SIFT
6. SURF and SURF

With the default parameters the first two detector-descriptor pairs using the Mikolajczyk's implementation of Hessian-affine detector were clearly superior to other methods [21], but here we adjust the meta-parameters to return the same average number of regions (300).

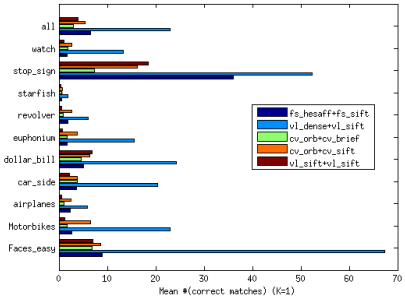
To these experiments, we also include the best fast detector-descriptor pair: ORB and BRIEF. The following combinations will be reported: *vl_sift+vl_sift* (FeatureSpace implementation), *fs_hessaff+fs_sift* (FeatureSpace implementation), *cv_orb+cv_brief* (OpenCV implementation), *cv_orb+cv_sift* (OpenCV, to compare SIFT and BRIEF), *vl_dense+vl_sift* (VLFeat implementation). We also tested the RootSIFT descriptor from [23] that achieved better performance in their experiments, but in our case it provided insignificant difference to the original SIFT (mean: $3.9 \rightarrow 4.2$, median: $1 \rightarrow 1$).

3.2. Performance measures and evaluation

In our preliminary work [21] we used a simplified version of the Mikolajczyk's descriptor performance measure: the ellipse overlap was replaced by normalized centroid distance of the matching regions. However, the results by the simplified rule turned out to be too optimistic and in this work we adopt the original measure. The rule is the same as with the detectors, if the best matching regions have sufficient overlap the match is counted correct. Descriptors are computed for all detected regions (foreground only). Images are processed pair-wise and the best match for each region is selected from the full distance matrix. It is worth noting that the rule proposed in [24] for discarding "bad regions" (ratio between the first and the second best is less than 1.5) is not used since it results complete failure. We used the ellipse overlap threshold 50% from [2], but also more strict thresholds were tested. Our performance numbers are the average number of matches and median number of matches. In the detector evaluation the mean and median numbers were almost the same, but here we report the both since for the descriptors there is significant discrepancies between the mean and median numbers.

3.3. Results

The average and median number of matches for the descriptor evaluation are shown in Fig. 5. For many classes, the mean and median numbers are very low and dense grid sampling is superior for all classes, achieving the average



Detector+descriptor	*Avg #*	*Med #*	*Avg # (60%)*	*(70%)*	*Comp. time (s.)*
vl_sift+vl_sift	3.9	1	2.8	1.6	0.15
fs_hessaff+fs_sift	6.5	2	5.9	4.9	0.22
vl_dense+vl_sift	23.0	10	22.3	20.2	0.76
cv_orb+cv_brief	3.0	1	2.9	2.7	0.11
cv_orb+cv_sift	5.4	2	4.8	4.1	0.37

Figure 5: Descriptor evaluation ($K = 1$ denotes the nearest neighbor matching, see Sec. 4.2 for more details). Top: average number of matches per class, Bottom: overall results table. The default overlap threshold is 50% [2], 60% and 70% results demonstrate the effect of the more strict overlaps. The computation times are average detector and descriptor computation times for one image pair.

of 23.0 and median of 10.0 matches. The more strict overlaps, 60% and 70%, provide almost the same numbers verifying that the matched regions do match well also spatially.

The best results were obtained for the stop signs, dollar bills and faces, but the overall performance is poor. The best discriminative methods could still learn to detect these categories, but it is difficult to imagine naturally emerging "common codes" for other classes except the three. It is surprising that the best detectors, Hessian-affine and dense sampling, provide on average 79 (192) corresponding regions, but only 10% of their descriptors match. The main

conclusion is that the descriptors that are developed for wide baseline matching do not work well in matching regions between different class examples.

3.4. The more the merrier?

Similar to Sec. 2.5 we study how the average number of matches behaves as the function of the number of extracted regions. This is justified as some works claim that "the more the merrier" [25]. The result graph is shown in Figure 6. The results show that adding more regions by adjusting the detector meta-parameters provides only minor improvement to the average number of matches. Clearly, the "best regions" are provided first and dense sampling performs much better indicating that what is "interesting" for the detectors is not necessarily a good object part.

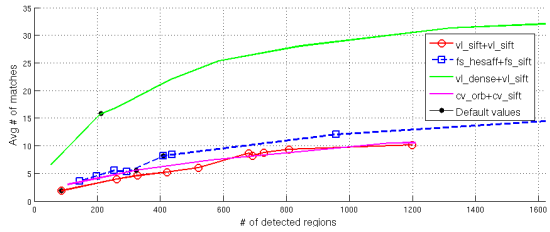
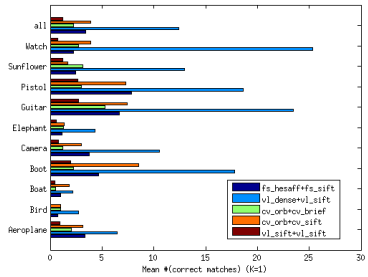


Figure 6: Descriptors' matches as functions of the number of detected regions controlled by the meta-parameters (default values denoted by black dots).

4. Advanced analysis

In this section, we address the open questions raised during the detector and descriptors comparisons in Section 2 and 3. The important questions are: why only a few matches are found between different class examples and what can be done to improve that? Why dense sampling outperforms all interest point detectors and does it have any drawbacks? Do our results generalize to other datasets?

4.1. ImageNet classes



(a)

Detector+descriptor	*Avg #*	*Med #*	*Avg # (60%)*	*(70%)*
vl_sift+vl_sift	1.2	0	0.7	0.3
fs_hessaff+fs_sift	3.4	2	2.8	1.9
vl_dense+vl_sift	12.4	7	11.6	10.2
cv_orb+cv_brief	2.2	1	1.9	1.5
cv_orb+cv_sift	3.9	2	3.3	2.5

(b)

Figure 7: Descriptor evaluation with the ImageNet classes to verify results in Fig. 5.

To validate our results, we selected 10 different categories from the the state-of-the-art object detection database: ImageNet [19]. The images were scaled to the same size as the Caltech-101 images and the foreground areas were annotated. The results for the ImageNet classes are in Figure 7. The average number of matches is roughly half of the number of matches with Caltech-101 images which can be explained by the fact that the dataset is more challenging due to 3D view point changes. However, the ranking of the methods is almost the same: dense sampling and SIFT is the best and SIFT detector and descriptor pair is the worst. The results validate our findings with Caltech-101.

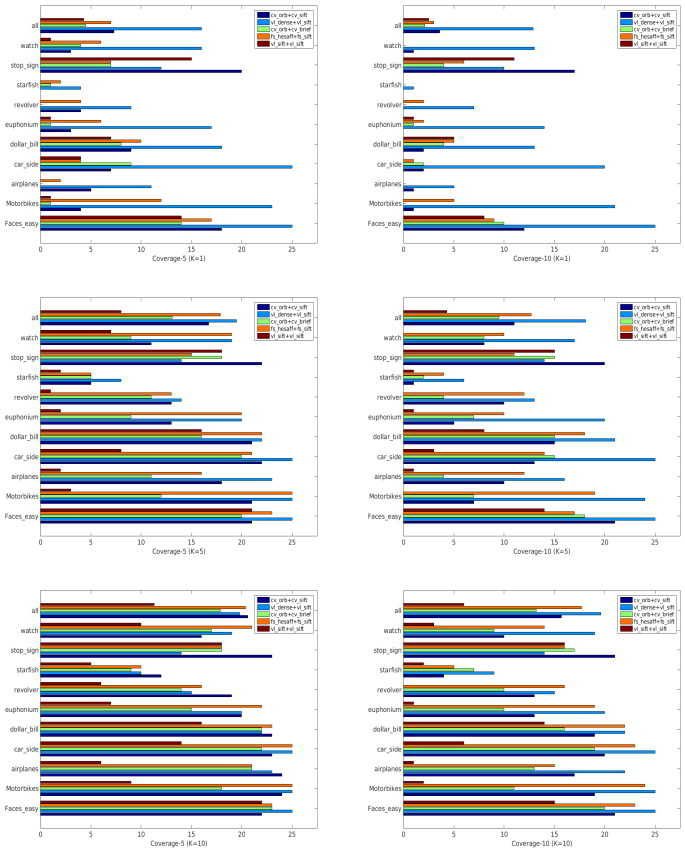


Figure 8: Number of image pairs for which at least $N = 5, 10$ (left,right) descriptor matches were found (*Coverage-N*). $K = 1, 5, 10$ denotes the number of best matches (nearest neighbors) counted in matching (top-down).

In object matching, assigning each descriptor to several best matches, "soft assignment" [26, 27, 28], provides improvement and we want to experimen-

Table 1: Average number of image pairs for which $N = 5, 10$ matches were found using $K = 1, 5, 10$ nearest neighbors.

	Coverage-($N = 5$)			*Coverage-($N = 10$)*		
Detector+descriptor	*K=1*	*K=5*	*K=10*	*K=1*	*K=5*	*K=10*
cv_orb+cv_sift	7.9	16.7	23.0	3.6	11.1	15.7
vl_dense+vl_sift	16.0	19.5	19.8	12.9	18.1	19.6
cv_orb+cv_brief	4.5	13.3	17.9	2.1	9.5	13.2
fs_hesaff+fs_sift	7.3	17.9	20.4	3.5	12.7	17.7
vl_sift+vl_sift	4.3	8.0	11.3	2.5	4.3	6.0

tally verify this finding using our framework. To measure the effect of multiple assignments, we establish a new performance measure: *coverage*. Coverage corresponds to the number of image pairs for which at least N matches have been found (*coverage-N*) and this measure is more meaningful than the average number of matches since there were strong discrepancies between the average and median numbers. We tested the multiple assignment procedure by accumulating matches over $n = 1, 2, \ldots, K$ best matches. The corresponding coverage for $K = 1, 5, 10$ are shown in Fig. 8 and Table 1. Obviously, more image pairs contain at least $N = 5$ than $N = 10$ matches. With $K = 1$ (only the best match) the best method, VLFeat dense SIFT, finds at least $N = 5$ matches (on average) in 16.0 out of 25 image pairs and 12.9 for $N = 10$. When the number of best matches is increased to $K = 5$, the same numbers are 19.5 and 18.1, respectively, showing clear improvement. Beyond $K = 5$ the positive effect diminishes and also the difference between the methods is less significant. Increase of the number of nearest neighbors in descriptor matching also makes performance gaps between the methods less significant.

4.3. Different implementations of the dense SIFT

During the course of work, we noticed that different implementations of the same method provided slightly different results. Since there are two popu-

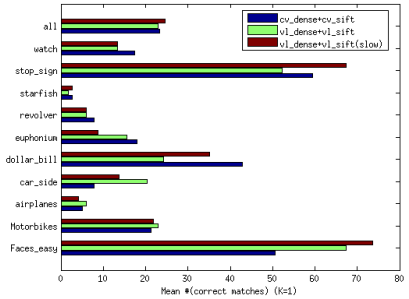


Figure 9: OpenCV dense SIFT vs. VLFeat dense SIFT (fast and slow) comparison.

lar implementations of dense sampling with the SIFT descriptor, OpenCV and VLFeat (two options: slow and fast), we compared them. The results corresponding to the previous experiments in Sec 3 are shown in Fig. 9. There are slight differences in classes due to implementation differences, but the overall performances are almost equal.

4.4. Challenging dense sampling: r-Caltech-101

Figure 10: The r-Caltech-101 versions of the original Caltech-101 images in Fig. 2 (original bounding box shown by green).

In dense sampling the main concern is its robustness to changes in scale and, in particular, orientation, since these are not estimated similar to interest point detection methods. In this experiment, we replicated the previous experiments with the two dense sampling implementations and the best interest point

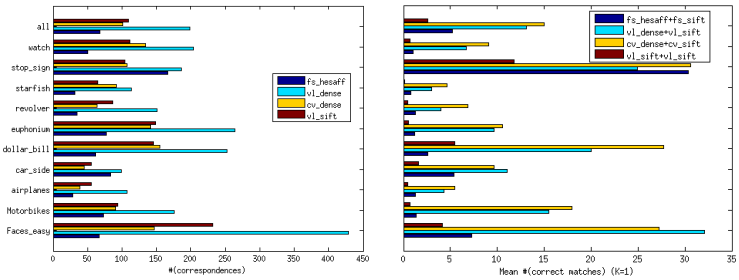


Figure 11: R-Caltech-101: detector (left) and descriptor (right). The detector results are almost equivalent to Fig. 3. In the descriptor benchmark (cf. with Fig. 5) the Hessian-affine performs better (mean: 3.4 $\rightarrow$ 5.2) while both dense implementations, VLFeat (23.0 $\rightarrow$ 13.1) and OpenCV (23.3 $\rightarrow$ 15.0) are severely affected.

detection method using the randomized version of the Caltech-101 data set, r-Caltech-101 [29]. R-Caltech-101 contains the same objects (foreground), but with varying random Google backgrounds and the objects have been translated, rotated and scaled randomly (Fig. 10).

The detector and descriptor results of this experiment are shown in Fig. 11 Now it is clear that artificial rotations affect the dense descriptors while Hessian-affine is unaffected (actually improves). It is noteworthy that the generated pose changes in r-Caltech-101 are rather small ($[-20°, +20°]$) and the performance drop could be more dramatic with larger variation. An intriguing research direction is detection of scaling and rotation invariant dense interest points.

5. Application: Image alignment

To verify our findings in a real application where region detectors and descriptors are core tools we selected the unsupervised feature-based object class image alignment method [30] for which state-of-the-art alignment accuracy is reported. The method takes as inputs an image ensemble and a single image

Table 2: Unsupervised image alignment accuracy with the feature-based congealing [30] for the original setting (Hessian-affine+SIFT) and the best in our evaluation: dense SIFT.

						
Orig. [30]	78%	53%	76%	27%	24%	2%
Orig. optim.	88%	86%	78%	35%	24%	4%
Dense orig.	96%	71%	86%	86%	20%	65%
Dense optim.	**98%**	**92%**	**90%**	**92%**	**53%**	**76%**

selected as a "seed". Matches between the seed and other images are computed and spatially matching seed descriptors accumulated over the process. The best seed descriptors are selected and all images are aligned using them. The process is simple, but depends on the success of the detector-descriptor pair. The original method uses the Hessian-affine detector and the SIFT descriptor with their default settings. The method's own meta-parameters are the normalized spatial match distance $\tau = 0.05$, the maximum number of seed landmarks $L = 20$, and the number of best descriptor matches $K = 10$.

The results are shown in Table 2 for the original detector-descriptor pair and for the vl_dense+vl_sift pair that performed best in our previous experiments. We used the same Caltech-101 classes from the previous experiments. During the experiments we found that the original parameter settings are sub-optimal and by cross-validation optimized them (Hessian-affine: $\tau = 0.02$, $L = 20$, $K = 2$; dense: $\tau = 0.04$, $L = 80$, $K = 10$). The performance number is the proportion of correctly aligned images measured by the normalized average distance of the annotated landmarks after alignment (0.10 corresponds to 10% of the distance between the two furthest landmarks - "object size"). In the both original and optimized settings the dense SIFT is clearly superior to the Hessian-affine and provides much better alignment performance even for the classes for which the original method performs poorly (airplanes and revolvers)

or fails (watches). See Figure 12 for alignment examples.

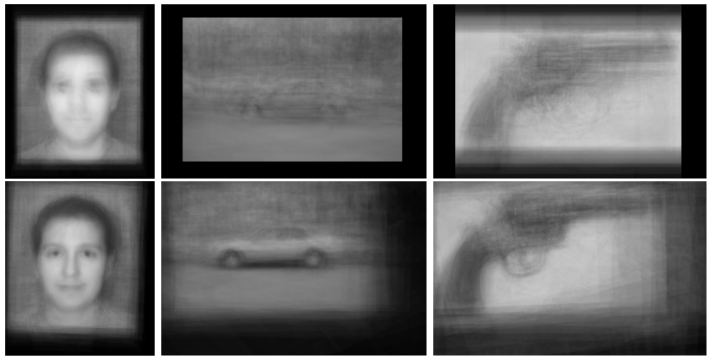


Figure 12: Average images without (top) and with unsupervised alignment (bottom).

6. Discussion

Interest points and regions have been the low-level features in visual class detection and classification for a decade [15]. Recently, supervised low-level features, such as convolution filters in deep neural networks [31], have gained momentum, but we believe that the unsupervised detector-descriptor approach can be developed further by identifying and improving the bottlenecks. In this work, we took a step to this direction by introducing an evaluation framework of part detectors and descriptors which provides intuitive and comparable results in the quantitative manner of the original works [1, 2].

With the proposed framework we identified the following important findings: 1) The original SIFT is the best descriptor (including the recent fast descriptors); 2) Dense sampling outperforms interest point detectors with a clear margin; 3) Detectors generally perform well, but descriptors' ability to match parts over visual class examples collapse; 4) Using multiple, even a few, best matches instead of the single best match provides significant performance boost; 5) Object pose

variation severely affects dense sampling while the best detector (Hessian-affine) is almost unaffected.

The findings advocate new research on i) optimization of the detector meta-parameters per visual class, ii) specialized descriptors for visual class parts and regions, iii) dense scaling and rotation invariant interest points, and iv) alternative matching methods for multiple best matches. Some results already exist. For example, BoW codebook descriptors can be enhanced by merging descriptors based on co-location and co-activation clustering [32] or by learning [33], dense interest points have been proposed [34], and soft-assignment has been shown to improve BoW codebook matching [26]. Moreover, the success of the standard SIFT in our experiments justifies further development of more effective visual class descriptors, not only more efficient descriptors. Investigating these potential research directions benefits from our evaluation framework that can be used for automatic validation and optimization.

References

[1] K. Mikolajczyk, T. Tuytelaars, , C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. V. Gool, A comparison of affine region detectors, Int J Comput Vis 65 (1/2) (2005) 43–72.

[2] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE PAMI 27 (10) (2005) 1615–1630.

[3] T. Tuytelaars, L. van Gool, Matching widely separated views based on affine invariant regions, Int J Comput Vis 1 (59).

[4] S. Se, D. Lowe, J. Little, Global localization using distinctive visual features, in: Int'l Conf. of Intelligent Robots and Systems, 2002, pp. 226–231.

[5] M. Brown, D. Lowe, Recognising panoramas, in: ICCV, 2003, pp. 1218–1227.

[6] H. Bay, A. Ess, T. Tuytelaars, L. V. Gool, Surf: Speeded up robust features, Computer Vision and Image Understanding (CVIU) 110 (3) (2008) 346–359.

[7] A. Alahi, R. Ortiz, P. Vandergheynst, FREAK: Fast retina keypoint, in: CVPR, 2012.

[8] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: an efficient alternative to SIFT or SURF, in: ICCV, 2011.

[9] S. Leutenegger, M. Chli, R. Siegwart, BRISK: Binary robust invariant scalable keypoints, in: ICCV, 2011.

[10] M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: Binary robust independent elementary features, in: ECCV, 2010.

[11] Z. Wang, B. Fan, F. Wu, Local intensity order pattern for feature description, in: ICCV, 2011.

[12] J. Zhang, M. Marszalek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: A comprehensive study, Int J Comput Vis 73 (2).

[13] K. Mikolajczyk, B. Leibe, B. Schiele, Local features for object class recognition, in: CVPR, 2005.

[14] C. Vondrick, A. Khosla, T. Malisiewicz, A. Torralba, HOGgles: Visualizing object detection features, in: ICCV, 2013.

[15] J. Sivic, A. Zisserman, Video google: A text retrieval approach to object matching in videos, in: ICCV, 2003.

[16] G. Csurka, C. Dance, J. Willamowski, L. Fan, C. Bray, Visual categorization with bags of keypoints, in: ECCV Workshop on Statistical Learning in Computer Vision, 2004.

[17] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part based models, IEEE PAMI 32 (9).

[18] L. Fei-Fei, R. Fergus, P. Perona, One-shot learning of object categories, IEEE PAMI 28 (4) (2006) 594.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database, in: CVPR, 2009.

[20] R. Hartley, A. Zisserman, Multiple View Geometry in computer vision, Cambridge press, 2003.

[21] J. Lankinen, V. Kangas, J.-K. Kamarainen, A comparison of local feature detectors and descriptors for visual object categorization by intra-class repeatability and matching, in: 21th Int. Conf. on Pattern Recognition (ICPR2012), 2012.

[22] M. Everingham, L. V. Gool, C. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results, http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html (2011).

[23] R. Arandjelovic, A. Zissermann, Three things everyone should know to improve object retrieval, in: CVPR, 2012.

[24] D. Lowe, Distinctive image features from scale-invariant keypoints, in: Int J Comput Vis, Vol. 60, 2004, pp. 91–110.

[25] E. Nowak, F. Jurie, B. Triggs, Sampling strategies for bag-of-features image classification, in: ECCV, 2006.

[26] A. Agarwal, B. Triggs, Multilevel image coding with hyperfeatures, Int J Comput Vis 78 (1).

[27] T. Tuytelaars, C. Schmid, Vector quantizing feature space with a regular lattice, in: ICCV, 2007.

[28] K. Chatfield, V. Lempitsky, A. Vedaldi, A. Zisserman, The devil is in the details: an evaluation of recent feature encoding methods, in: BMVC, 2011.

[29] T. Kinnunen, J.-K. Kamarainen, L. Lensu, J. Lankinen, H. Kälviäinen, Making visual object categorization more challenging: Randomized Caltech-101 data set, in: 20th Int. Conf. on Pattern Recognition (ICPR2010), 2010.

[30] J. Lankinen, J.-K. Kamarainen, Local feature based unsupervised alignment of object class images, in: BMVC, 2011.

[31] A. Krizhevsky, I. Sutskever, G. Hinton, ImageNet classification with deep convolutional neural networks, in: NIPS, 2012.

[32] B. Leibe, A. Ettlin, B. Schiele, Learning semantic object parts for object categorization, Image and Vision Computing 26 (2008) 15–26.

[33] K. Simonyan, A. Vedaldi, A. Zisserman, Learning local feature descriptors using convex optimisation, IEEE PAMI 36 (8).

[34] T.Tuytelaars, Dense interest points, in: CVPR, 2010.

PUBLICATION

II

Robustifying correspondence based 6D object pose estimation

A. Hietanen, J. Halme, A. G. Buch, J. Latokartano and J.-K. Kämäräinen

*International Conference on Robotics and Automatio*2018, 739–745

Robustifying Correspondence Based 6D Object Pose Estimation

Antti Hietanen[1], Jussi Halme[2], Anders Glent Buch[3], Jyrki Latokartano[2] and J.-K. Kämäräinen[1]

***Abstract*— We propose two methods to robustify point correspondence based 6D object pose estimation. The first method, curvature filtering, is based on the assumption that low curvature regions provide false matches, and removing points in these regions improves robustness. The second method, region pruning, is more general by making no assumptions about local surface properties. Our region pruning segments a model point cloud into cluster regions and searches good region combinations using a validation set. The robustifying methods are general and can be used with any correspondence based method. For the experiments, we evaluated three correspondence selection methods, Geometric Consistency (GC) [1], Hough Grouping (HG) [2] and Search of Inliers (SI) [3] and report systematic improvements for their robustified versions with two distinct datasets.**

I. INTRODUCTION

6-DoF object pose estimation from 3D data (point cloud/colored point cloud) is an active yet challenging problem in robotics, e.g., for vision based manipulation [4], [5]. A popular approach is to find correspondence point between the captured scene and stored models which are both represented by 3D point clouds [1], [2], [3]. It turns out that in practice these methods can easily fail if an object is observed from a difficult view point, or if other objects occlude a large part of the object. In this work, we assume that not all points can be treated with equal importance, e.g. large solid areas and sharp object corners, but robustness can be improved by selecting a good sub-set of points that guarantees more robust pose estimation (see Fig. 1). Our research problem is to identify a good sub-set of the model points.

In this work, we assume availability of a set of validation images that represent typical scene captures and propose two methods to robustify correspondence based pose estimation. The first method is based on our findings of failures cases in automated heavy outdoor robot tool changing, where many tools contain large planar areas that provide false correspondences, consequently leading to poor pose estimates. To remove planar areas we exploit computational curvature estimates and filter out low curvature regions. The second method does not make assumptions about the shape properties around surface points, but divides the model point cloud into local regions by clustering. Then a randomized procedure is executed to find a good combination of these regions. Our experiments with two distinct datasets verify that our robustifying procedures consistently achieve better accuracy and decrease the number of wrong or inaccurate pose estimates.

[1]Signal Processing Laboratory, [2]Laboratory of Mechanical Engineering and Industrial Systems, Tampere University of Technology
[3] SDU Robotics, University of Southern Denmark

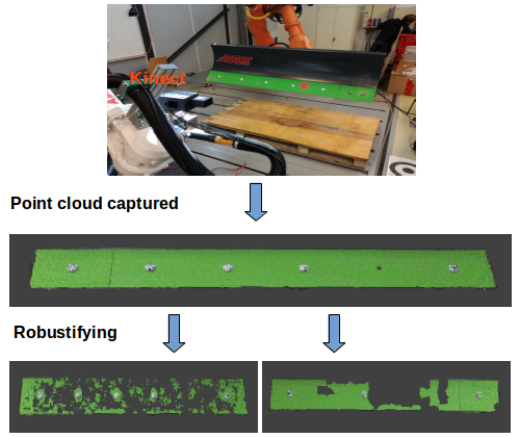


Fig. 1. Robot setup of a fixed RGB-D sensor (Kinect) and a snow blade attached to a manipulator (top). 3D colored point cloud of the green part of the blade containing attaching bolts (middle). Robust sub-sets of the points by Curvature Filtering (bottom left) and Region Pruning (bottom right).

II. RELATED WORK

In the following we focus on 3D-to-3D pose estimation methods and, in particular, methods that store object models and capture test scenes as 3D point clouds. Many proposed methods are developed for object recognition, but since they are also suitable for pose estimation we include them here.

3D Object Recognition – Local region detectors and descriptors have been successful in 2D vision problems and have therefore been extended to 3D surfaces and point clouds, e.g., 3D SURF [6], 3D HOG and DoG [7]. A recent survey and evalutaions of the detectors and descriptors can be found in Guo et al. [8], [9]. The descriptors provide reliable object recognition, but for accurate pose estimation the best result can be achieved by registering model points to corresponding scene points. For this registration process, obtaining correct point correspondences becomes a crucial task.

Point cloud based methods have been proposed by Papazov and Burschka [10] and Drost et al. [11]. Papazov and Burschka utilise a random sample consensus (RANSAC) matching and Drost et al. use Hough-like voting. More recently, a mesh-based local descriptor was used for achieving good results for a series of 3D recognition tasks [12]. In another work [13], local descriptors were integrated into a sophisticated global hypothesis verification framework. Re-

cently, attention has also been paid to 3D point selection [14], [3], and in this work we adopt three recent methods with distinctive approaches: Geometric Consistency (GC) [1], Hough Grouping (HG) [2] and combined local and global Search of Inliers (SI) [3]. We describe these three selected methods in more details in Section III.

6D Pose Estimation – A 3D point cloud is the typical modality used for object pose estimation in robotics [4], [5]. It is noteworthy that the best RGB-D SLAM methods are also based on point clouds [15], [16], but in their case the previous frame provides a good initial estimate of the pose and can be refined by dense gradient or Iterative Closest Point (ICP) matching. In the case of object pose estimation, the initial estimate for the ICP must be provided by robust correspondence-based estimation (RANSAC, Hough Voting) [10], [11] that cope with occlusion and clutter. In the most recent works, more complex correspondence search algorithms have been proposed that utilize the neighborhoods of the surface points [4], [3], [5].

Contributions – We propose two methods to robustify correspondence based 6D object pose estimation

- *Curvature Filtering* that removes points within low curvature areas of model point clouds and
- *Region Pruning* that processes the model point cloud as local regions for which a good combination is sought using a trial-and-error procedure with validation data.

Effectiveness of the proposed robustifying methods is verified using two distinct datasets where the first one is used in many related works and the second one is generated by ourselves using tools for our outdoor robot for land moving and snow clearance. Our dataset, ground truth and code will be made publicly available.

III. 3D CORRESPONDENCE METHODS

For baseline methods in this work, we selected tree recent methods available: *Geometric Consistency* (GC) [1], *Hough Grouping* (HG) [2] and *Search of Inliers* (SI) [3]. In our experiments, for GC and HG we use the available implementations in the Point Cloud Library [17] and for SI we use the implementation by the original authors. In the following, we briefly explain these methods and their most important parameters.

All three methods start processing initial correspondence candidates and refine the model using various correspondence verification procedures that remove poor matches between two point clouds (a model and query scene). The initial correspondence are created by using the SHOT features [18] that performed well in the recent comparison [9] and provide good balance between computational complexity and performance. Fast nearest neighbor search for initial correspondence is done using the FLANN library (Fast Library for Approximate Nearest Neighbors [19]).

A. Search of Inliers (SI) [3]

The SI method is based on two consequtive processing stages, *local voting* and *global voting*. At the end the votes are accumulated to form a quantitative indicator (number of votes) to denote the degree of trust for each correspondence.

The initial corresponses are refined by Lowe's test of ratio between the best and the second best matches with the threshold set to $\geq \tau_{Lowe} = 0.2$. The ratio test refines the original set of correspondences $\mathcal{C}$ to a sub-set $\mathcal{C}_{Lowe}$ such that $|\mathcal{C}_{Lowe}| \leq |\mathcal{C}|$. The first voting step performs local voting, where locally selected correspondence pairs are selected from the model and a scene, and the score is computed using their pair-wise similarity score $s_{local}(\vec{p})$ for each 3D point $\vec{p}$. The second global voting stage samples point correspondences, estimates a transformation and gives a global score to the points correctly aligned outside the estimation point set: $s_G(\vec{p})$. The final score $s(\vec{p})$ is computed by integrating both local and global scores, and an adaptive threshold between inliers and outliers is automatically found by Otsu's bimodal distribution thresholding. The inlier set is used for final pose estimation. The two following methods alter the input correspondence set $\mathcal{C}$ by selecting only robust sub-regions such that $\mathcal{C}' \subseteq \mathcal{C}$.

B. Geometric Consistency (GC) [1]

The geometric consistency (GC) incrementally builds regions (clusters) of correspondence that are geometrically consistent. In our work we will use the implementation by Chen and Bhanu [1] which is a modified version of the original method by Johnson and Hebert [20], [21] and available in Point Cloud Library [17].

The method clusters correspondence pairs of similar accuracy by imposing an absolute pairwise distance constrained equal to the Euclidean distance between the feature points:

$$\Big| \|p_{i,m} - p_{j,m}\|^2 - \|p_{i,s} - p_{j,s}\|^2 \Big| < \tau_\epsilon, \qquad (1)$$

where $p_{:,m}$:s are model points and $p_{:,s}$ captured scene points. We initialize the algorithm with $|\mathcal{C}|$ clusters each having a seed correspondence. Then for each cluster we search the set of correspondence whose pairwise distances are less than a predefined threshold value τ_ϵ. The clustered correspondences are marked as visited, and the seed growing repeats until all correspondences have been visited. As the final step, RANSAC can be computed to refine each cluster set [22].

In principle, GC can return more than one correspondence cluster and for pose estimation we have to rank them using a suitable metric. In our evaluations, we found that the cluster which has the largest number of correspondence lead to best average pose estimation.

C. Hough Grouping (HG) [2]

The key idea of the Hough 3D correspondence grouping (HG) [2] is to iteratively cast votes for object location and pose bins in the Hough parameter space and at the end of the process the highest accumulated bins represent the most likely pose candidates and correspondence contributed to the bins are accepted.

The method requires an unique reference point in the model, typically the model centroid, and each bin represents

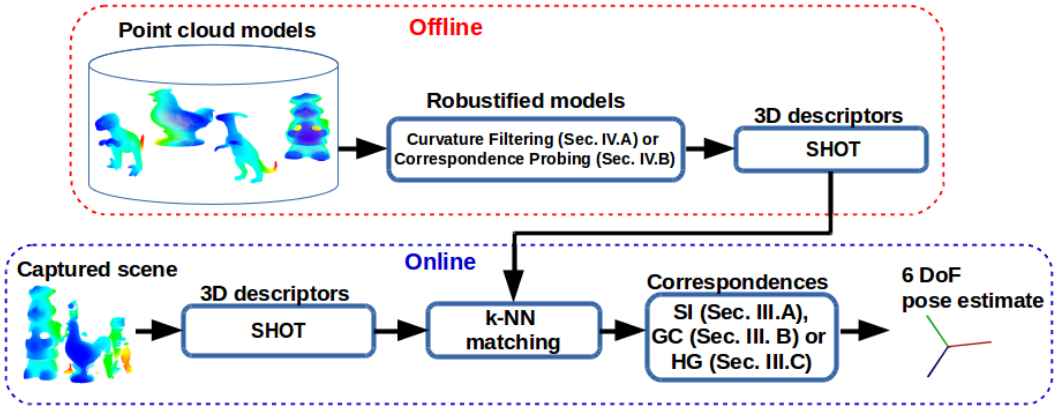


Fig. 2. Used model for correspondence based 6D object pose estimation.

a single pose instance candidate. Therefore correct correspondence vote a same bin which gets quickly accumulated. To make correspondence points invariant to rotation and translation between the model and scene, every point is associated with local Reference Frame (RF) [18]. In the voting stage each correspondence between a capture scene and a model cast a single vote to a single or multiple bins in the 3D translation Hough accumulator space and pose is stored in the local reference frame. Finally, correspondence contributing bins having votes more than a set threshold which is adaptively set as it depends on the number of available points and the most important parameter is the Hough accumulator bin size. In addition, in [23] two different weighting methods for the voters were proposed, but in our experiments they did not improve performance.

IV. ROBUSTIFYING METHODS

In the following, we explain the two proposed methods to robustify pose estimation with more reliable correspondences.

A. Curvature Filtering

Curvature is surface property that may affect to 3D object detection, tracking and pose estimation. For example, tracking does not converge on large planar areas where matches can be equally good everywhere. On the other hand, sudden surface normal changes in high curvature areas, such as corners and edges, provide strong cues for tracking and pose estimation. There also exist a number of studies on perceptual experiments that demonstrate the importance of curvature in the human visual system [24], [25].

We compute the curvature value of a point as the *surface variation* defined in [26]:

$$\sigma = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \ , \tag{2}$$

where the λ:s are the eigenvalues of the corresponding eigen vectors $\vec{v}_i$ (λ_0 is the largest) of the covariance matrix C:

$$C = \frac{1}{N_{curv}} \sum_{i=1}^{N_{curv}} (\vec{p}_i - \vec{\mu}) \cdot (\vec{p}_i - \vec{\mu})^T \ , \tag{3}$$

where N_{curv} is the number of points considered in the neighbourhood of $\vec{p}_i$, and $\vec{\mu}$ represents the 3D centroid (mean) of the points.

The number of neighbours is a free parameter of the method but it should be set large enough to tolerate noise. The second free parameter of the method is the actual curvature threshold, which we denote τ_{curv}. Points having lower curvature value than τ_{curv} will be removed from the point cloud. Figure 1 illustrates the model after curvature based selection.

B. Region Pruning

First we segment the model point cloud to supervoxels (Figure 3) using the algorithm described in [27]. The grouping starts by dividing the 3D space of the model into a voxelized grid with resolution R_{seed}. Expansion of the supervoxels is then done by local k-means clustering controlled by the feature distance measure:

$$D = \sqrt{w_c D_c^2 + \frac{w_s D_s^2}{3R_{seed}} + w_n D_n^2}, \tag{4}$$

where D_s is the spatial distance by the seeding resolution, D_c is the Euclidean color distance in normalized RGB space, and the normal distance D_n measures the angle between surface normal vectors. Weights w_c, w_s and w_n control the influence of color, spatial and normal features respectively. Finally we end up with n supervoxels each having a central point $p_n(x, y, z)$.

Now the main task is to "prune" the generated regions (see Figure 3) and select a good sub-set that provides the best performance using a validation set. It is apparent that for a

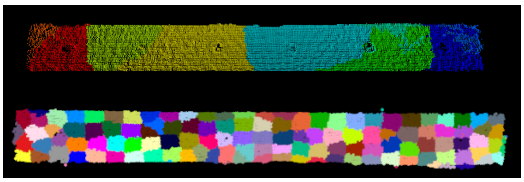


Fig. 3. Top: snow blade point cloud divided to 10 supervoxels; Bottom: 128 supervoxels.

large k the exhaustive search of the best combination quickly becomes computationally intractable. Exhaustive search with k regions requires

$$\frac{n!}{(n-k)!k!} \tag{5}$$

experiments with all validation set scenes. The total number of tests is

$$\sum_{k=1}^{n} \frac{n!}{(n-k)!k!} \tag{6}$$

combinations which is infeasible except for a very small n (10-15). The solution used in this work is to perform random pruning of 1-10% of the regions with a fixed R_{seed} and k. This procedure is experimentally evaluated in Section V-D.

V. EXPERIMENTS

In this section, we report results for the experiments with the three correspondence methods (GC, HG and SI - see Section III) combined with the proposed robustifying methods in Section IV: *Curvature Filtering* (curv) and *Region Pruning* (regp). We also provide results for the correspondence methods with validation set optimized parameters (GC-opt, HG-opt and SI-opt) and using RANSAC as the standard robustifying procedure.

A. Data and Performance Measure

Laser Scanner Dataset – As a benchmark to compare to other works we use the Laser Scanner Dataset[1], which has been used to evaluate 3D object recognition and 6D pose estimation methods [28], [2], [3]. The dataset contains four difficult models: *T-rex*, *Chicken*, *Parasaurolophus* and *Cheff*. Objects are occluded in the test scenes and in average $71\% - 77\%$ of the points are missing. The dataset contains also ground truth transformation matrices to align each model to the test scenes.

Outdoor Robot Tool Dataset – The tool dataset was collected using our robot setup (Figure 4) where a ABB IRB6640 manipulator was used to systematically move the selected tools (a snow blade and a container box) to different locations and pose angles. Each configuration was captured by a Kinect v2 sensor. One of the views was selected as the canonical view and for all other views we provide $4x4$ homogeneous transformation matrices that align them to the canonical view. The groundtruth transformation matrices were generated by manually selecting corresponding points in all point clouds and using the direct linear method for initial estimation and the Iterative Closest Point (ICP) algorithm to refin estimation [29]. RGB-D images, ground truth transformations and our evaluation code will be made publicly available.

[1] http://staffhome.ecm.uwa.edu.au/ 00053650/recognition.html

Fig. 4. The two outdoor robot tools used in our dataset: a container box (left) and a snow blade (180 kg). An ABB manipulator was used to systematically change the view point and RGB-D data was recorded using a Kinect V2 on a tripod.

Error measure – We adopt the error measure proposed in [3], which measures the mean squared error (MSE) of all model points $\vec{p} \in M$ using the ground truth transformation $\mathcal{T}_{gt}$ and the estimated transformation $\hat{\mathcal{T}}$:

$$\epsilon_{MSE} = \frac{1}{|M|} \sum_{\vec{p} \in M} \|\hat{\mathcal{T}}(\vec{p}) - \mathcal{T}_{gt}(\vec{p})\|^2 \ . \tag{7}$$

Since some methods may completely fail for certain test scenes, we also report top-50% and top-25% MSE values, which are less affected by estimation failures providing large errors.

B. Method Comparison

The results for the selected three methods and their variants are shown in Table I for the Laser Scanner Dataset and in Table II for our Outdoor Robot Tool dataset. From the results we can make the following observations: the Geometric Consistency (GC) based correspondence provide the most accurate and robust pose estimation. For the Laser Scanner Dataset objects GC variants are the best for 10/12 cases and SI-opt (curv) wins 2/12.

In general, parameter optimization with a validation dataset always improves accuracy; this is particularly evident for top-50% and top-25% MSEs indicating that fewer poses are falsely detected (far from the true pose). For GC, RANSAC post-processing sometimes improves the results, but for HG most of the time it does not. The curvature based filtering to robustify the methods does not improve HG and GC, but consistently improves SI making it comparable or even better than HG and GC. Region pruning consistently improves both GC and SI often achieving the best accuracy (8 out of 12 cases).

For our own dataset in Table II the results are very similar, although the objects are very different from those in the Laser Scanner dataset - our objects contain many large planar areas which supposedly should benefit from curvature filtering. Again Geometric Consistency (GC) variant is the winning

TABLE I

METHOD PERFORMANCE FOR THE LASER SCANNER DATASET. **Note:** *RANSAC IS PART OF THE METHOD.

	Cheff			T-rex			Chicken			Parasaurolophus		
$\times 10^{-3}$	MSE	top-50%	top-25%	MSE	top-50%	top-25%	MSE	top-50%	top-25%	MSE	top-50%	top-25%
Original with default parameters												
GC [1]	6.235	0.026	0.002	24.111	9.437	0.479	12.997	2.188	0.070	50.493	3.091	0.012
HG [2]	45.719	26.425	22.599	62.805	34.913	21.927	13.633	3.233	0.227	51.331	9.464	1.907
SI [3]	15.622	0.049	0.034	24.906	12.904	4.858	16.552	3.360	0.111	46.051	3.588	0.012
Optimized parameters												
GC-opt	5.108	**0.002**	**0.001**	**17.321**	8.015	0.197	11.175	1.727	0.042	46.253	2.697	0.009
HG-opt	7.586	0.520	0.003	17.557	12.496	7.334	10.592	2.829	0.227	46.772	8.679	0.624
SI-opt	5.300	0.021	0.010	27.700	11.600	3.900	13.000	1.678	0.012	46.000	3.503	0.005
Optimized & RANSAC												
GC-opt-RANSAC	3.100	0.006	0.001	19.464	7.150	0.102	10.936	1.554	0.089	48.000	2.575	0.016
HG-opt-RANSAC	35.501	21.260	15.000	45.900	21.100	15.200	14.396	2.484	0.204	46.981	7.572	0.049
SI-opt*	5.300	0.021	0.010	27.700	11.600	3.900	13.000	1.678	0.012	46.000	3.503	0.005
Our robustifying procedures												
GC-opt-RANSAC (curv)	6.900	0.036	0.010	21.440	13.838	7.779	10.537	1.581	0.100	45.600	2.720	0.024
HG-opt-RANSAC (curv)	13.930	5.207	2.937	21.881	9.711	3.198	12.374	1.989	0.179	50.663	3.852	0.141
SI-opt (curv)	3.900	0.017	0.007	21.900	8.385	0.384	10.017	1.252	**0.007**	45.900	2.963	**0.002**
GC-opt (regp)	**2.332**	0.004	**0.001**	18.326	**5.320**	**0.050**	**9.301**	**0.779**	0.016	**45.198**	**1.952**	0.006
HG-opt (regp)	14.670	9.718	9.134	32.136	21.249	15.897	29.902	15.622	11.512	60.179	22.303	16.045
SI-opt (regp)	4.600	0.018	0.007	22.600	8.300	1.300	16.734	3.496	0.120	46.695	3.131	0.082

TABLE II

METHOD PERFORMANCE FOR THE OUTDOOR ROBOT TOOL DATASET.

	Blade			Box		
	MSE	top-50%	top-25%	MSE	top-50%	top-25%
GC [1]	6.2730	1.0320	0.1890	6.7880	2.6190	0.0002
HG [2]	6.0620	0.6960	0.1240	8.7000	5.4800	1.5330
SI [3]	2.4080	0.0200	0.0005	9.7780	4.9950	0.0389
GC-opt	0.8671	**0.0003**	**0.0001**	5.7827	1.6394	**0.0001**
HG-opt	4.6077	0.2024	0.0510	6.7606	3.1600	0.0002
SI-opt	2.1690	0.0022	0.0005	6.3588	3.0081	0.0005
GC-opt-RANSAC	0.4184	0.0004	0.0002	4.1384	0.0463	0.0002
HG-opt-RANSAC	0.7333	0.2010	0.0330	6.0916	1.7224	**0.0001**
SI-opt	2.1690	0.0022	0.0005	6.3588	3.0081	0.0005
GC-opt-RANSAC (curv)	**0.2280**	0.0004	0.0002	**2.9893**	**0.0113**	**0.0001**
HG-opt-RANSAC (curv)	0.2595	0.1288	0.0334	6.0283	0.0249	0.0002
SI-opt (curv)	2.1734	0.0020	0.0004	6.2161	1.4291	0.0004
GC (regp)	0.2744	**0.0003**	0.0002	5.1058	0.1475	0.0002
HG (regp)	2.2614	0.2367	0.0805	7.4540	2.8303	0.0006
SI-opt (regp)	2.2948	0.0014	0.0007	6.0578	2.0800	0.0014

method in all cases (6/6). Clearly, the method of choice is GC with optimized parameters and curvature filtering as the GC-opt-RANSAC (curv) wins 4/6 cases.

C. Curvature Filtering

In the method comparison experiments (results in Tables I and II) robustifying was performed by optimizing the curvature filtering parameters with a validation set (example scenes). As described in Section IV-A the two important parameters are the *number of neighbour points* N_{curv} to estimate the curvature value and the *curvature threshold* τ_{curv}, which is used to remove low curvature points (below the threshold). MSEs using varying values of the curvature parameters for the snow blade images are shown in Figure 5 and Figure 6. We can make two observations: the neighbourhood size must be large enough to compute a robust curvature estimate (≥ 5). However, finding a suitable value for the curvature threshold is essential and is likely to depend on each model's properties. One should also note that although curvature filtering does not significantly improve GC or HG, we can still remove insignificant points while maintaining the same or an even better pose estimation rate.

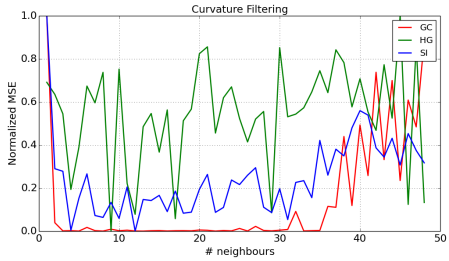


Fig. 5. Effect of the neighborhood size N_{curv} parameter to the performance of curvature filtering (snow blade).

D. Region Pruning

The good property of the Region Pruning method (Section IV-B) is that it does not make assumption on what kind of point cloud regions are good for robust pose estimation. The main parameter of the region pruning is the number of regions N_{reg} which also defines the computational time and it turns out that exhaustive search is doable only for $N_{reg} \leq 10$, but for good results we typically need $N_{reg} \geq 100$. In our case this was solved by randomly removing 10% of the regions and executing this random procedure 1,000 times.

E. Optimizing Method Parameters

From Table I and Table II it is clear that each method's parameters affect to the performance and robustify the method if individually set for each object.

Search of Inliers (SI) – The main parameters of the SI method are related to its two voting stages: local voting and global voting. The parameters that strongly influence the performance are the size of local voting neighbourhood

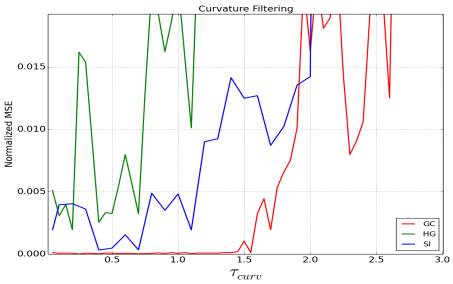


Fig. 6. Effect of the curvature threshold τ_{curv} parameter to the performance of curvature filtering (snow blade).

(the default value is 250) and the correspondence distance of global voting (the default value ≥ 0.9). The pose estimation errors as functions of the two parameters are shown in Figure 7. The default values perform reasonably well for the neighbourhood size, but the effect of the correspondence distance is important for robustness. For the snow blade, the optimal values are far from the default settings and there are optimal points for a low values 0.1 and high value ≥ 0.92 which indicates "alternative" point regions for robust pose estimation and these settings can only be found using cross-validation.

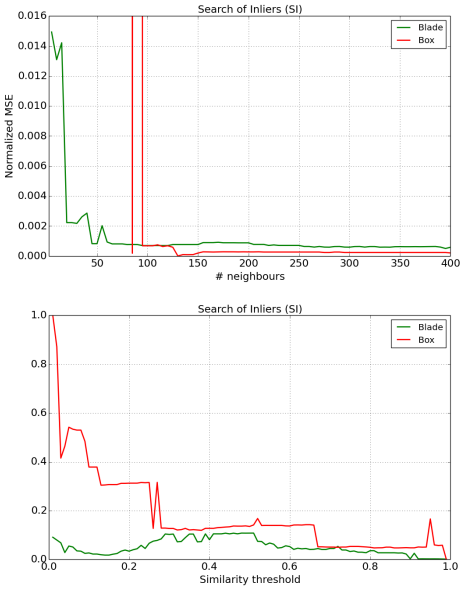


Fig. 7. Snow blade estimation error as the function of the SI parameters: neighborhood size (top) and distance threshold (bottom)

Geometric Consistency (GC) – The main parameter for GC is the geometrical consistency threshold and the results from the parameter optimization are shown in Figure 8. We can see that the optimal pairwise distance between correspondence points is 7 mm for the snow blade. The value is approximately $2\times$ higher than the threshold value used with the laser data. This is understandable due to Kinect's noisy sensor data.

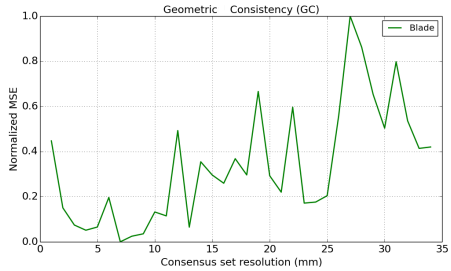


Fig. 8. Snow blade estimation error as the function of the GC consistency threshold.

Hough Grouping (HG) – The main parameter for HG is the Hough accumulation space bin size and the results from the parameter optimization are shown in Figure 9. A good value for the bin size is approx. 4 mm for the snow blade object.

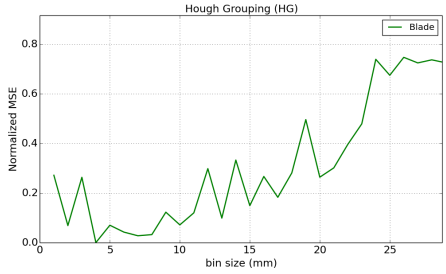


Fig. 9. Snow blade estimation error as the function of the Hough space's bin size.

VI. CONCLUSIONS

We proposed two alternative methods to improve 3D point correspondence based object pose estimation. Our methods, Curvature Filtering (Section IV-A) and Region Pruning (Section IV-B), were used to select a robust sub-set of correspondence against estimation failures. In our experiments (Table I and Table II), the robustifying methods consistently improved the three correspondence based methods: Geometric Consistency (GC) [1], Hough Grouping (HG) [2] and Search of Inliers (SI) [3]. Surprisingly, in all experiments Geometric Consistency (GC) outperformed the other two as combined with our robustifying using Region Pruning (Laser Scanner Dataset) or Curvature Filtering (Outdoor Robot

Tool Dataset). There was no clear winner between the two robustifying methods and more work is required to find the most suitable one. Our future work will address combining the two robustifying methods, branch-and-bound search for faster region pruning and cross-validation without validation images, i.e. using rendered views of the model point cloud itself. Robustified GC will be used in an autonomous robot service station where outdoor robot can change its tool.

ACKNOWLEDGMENT

The research leading to these results has received funding from the Tampere University of Technology Robotics and Intelligent Machines Flagship project and the Academy of Finland (the ROSE project under the grant 292980).

REFERENCES

[1] H. Chen and B. Bhanu, "3d free-form object recognition in range images using local surface patches," *Pattern Recogn. Lett.*, vol. 28, pp. 1252–1262, July 2007.

[2] F. Tombari and L. Di Stefano, "Object recognition in 3d scenes with occlusions and clutter by hough voting," in *PSIVT*, pp. 349–355, IEEE, 2010.

[3] A. Buch, Y. Yang, N. Krüger, and H. Petersen, "In search of inliers: 3d correspondence by local and global voting," in *CVPR*, 2014.

[4] A. Buch, D. Kraft, J.-K. Kamarainen, H. Petersen, and N. Krüger, "Pose estimation using local structure-specific shape and appearance context," in *ICRA*, 2013.

[5] C. Li, J. Bohren, E. Carlson, and G. Hager, "Hierarchical semantic parsing for object pose estimation in densely cluttered scenes," in *ICRA*, 2016.

[6] J. Knopp, M. Prasad, G. Willems, R. Timofte, and L. van Gool, "Hough transform and 3D SURF for robust three dimensional classification," in *ECCV*, 2010.

[7] A. Zaharescu, E. Boyer, and R. Horaud, "Keypoints and local descriptors of scalar functions on 2d manifolds," *IJCV*, vol. 100, pp. 78–98, 2012.

[8] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, "3d object recognition in cluttered scenes with local surface features: A survey," *PAMI*, vol. 36, no. 11, 2014.

[9] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. Kwok, "A comprehensive performance evaluation of 3D local feature descriptors," *IJCV*, vol. 116, pp. 66–89, 2016.

[10] C. Papazov and D. Burschka, "An efficient RANSAC for 3D object recognition in noisy and occluded scenes," in *ACCV*, 2010.

[11] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *CVPR*, 2010.

[12] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3d local surface description and object recognition," *IJCV*, vol. 105, no. 1, pp. 63–86, 2013.

[13] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, "A global hypotheses verification method for 3d object recognition," in *ECCV*, pp. 511–524, 2012.

[14] E. Rodola, A. Albarelli, F. Bergamasco, and A. Torsello, "A scale independent selection process for 3d object recognition in cluttered scenes," *IJCV*, vol. 102, no. 1, pp. 129–145, 2013.

[15] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *ISMAR*, pp. 127–136, 2011.

[16] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald, "Kintinuous: Spatially extended KinectFusion," in *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, (Sydney, Australia), Jul 2012.

[17] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *ICRA*, (Shanghai, China), May 9-13 2011.

[18] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *ECCV*, (Berlin, Heidelberg), pp. 356–369, Springer-Verlag, 2010.

[19] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *PAMI*, vol. 36, 2014.

[20] A. E. Johnson and M. Hebert, "Surface matching for object recognition in complex 3-d scenes," *Image and Vision Computing*, vol. 16, pp. 635–651, 1998.

[21] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *TPAMI*, vol. 21, pp. 433–449, May 1999.

[22] A. Aldoma, F. Tombari, L. di Stefano, and M. Vincze, "A global hypotheses verification method for 3d object recognition.," in *ECCV* (A. W. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, eds.), vol. 7574 of *Lecture Notes in Computer Science*, pp. 511–524, Springer, 2012.

[23] S. Salti, F. Tombari, and L. di Stefano, "On the use of implicit shape models for recognition of object categories in 3d data.," in *ACCV* (R. Kimmel, R. Klette, and A. Sugimoto, eds.), vol. 6494 of *Lecture Notes in Computer Science*, pp. 653–666, Springer, 2010.

[24] F. Attneave, "Some informational aspects of visual perception.," *Psychol Rev*, vol. 61, no. 3, pp. 183–193, 1954.

[25] I. Biederman, "Recognition-by-components: A theory of human image understanding," *Psychological Review*, vol. 94, pp. 115–147, 1987.

[26] M. Pauly, M. Gross, and L. P. Kobbelt, "Efficient simplification of point-sampled surfaces," in *Proceedings of the conference on Visualization'02*, pp. 163–170, IEEE Computer Society, 2002.

[27] J. Papon, A. Abramov, M. Schoeler, and F. Wörgötter, "Voxel cloud connectivity segmentation - supervoxels for point clouds," in *CVPR*, (Portland, Oregon), June 22-27 2013.

[28] A. Mian, M. Bennamoun, and R. Owens, "Three-dimensional model-based object recognition and segmentation in cluttered scenes," *PAMI*, vol. 28, no. 10, 2006.

[29] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vision Comput.*, vol. 10, pp. 145–155, Apr. 1992.

PUBLICATION III

Depth-sensor–projector safety model for human-robot collaboration

A. Hietanen, R.-J. Halme, J. Latokartano, R. Pieters, M. Lanz and J.-K. Kämäräinen

*International Conference on Intelligent Robots and Systems Workshop on Robotic Co-workers 4.0*2018

Depth-sensor–projector safety model for human-robot collaboration

Antti Hietanen[1], Roni-Jussi Halme[2], Jyrki Latokartano[2], Roel Pieters[3], Minna Lanz[2] and Joni-Kristian Kämäräinen[1]

***Abstract*— We propose a depth sensor and projector based safety model for human-robot collaboration in a compact shared workspace. The model consists of three spatial zones, *robot zone*, *danger zone* and *human zone*. The zones are online modelled, updated and monitored using a single depth sensor and user notification and interaction is provided by a projector-mirror display. Our model includes methods for detection of safety zone violations (an obstacle enters the danger zone) and prevents the robot to move to "unverified" (changed) workspace regions. Unverified regions are verified by user interaction. In the experimental part, we define an assembly task with the standard Cranfield benchmark parts where our proposed model reduces robot idle time by 43% and achieves 12.4% average reduction in task completion time as compared to a baseline.**

I. INTRODUCTION

Human-robot interaction (HRI) for collaborative manufacturing requires special attention for HRI safety systems since heavy robots and payloads can lead to potentially dangerous situations. In addition to be effective and efficient, the HRI safety systems should be affordable and easy-to-install to be easily deployable. In this work, we focus on depth sensor and projector based safety in a shared workspace for collaborative manufacturing (HRI based assembly and disassembly) since this hardware is affordable, easy to install and re-configure.

Previous works on vision/depth-based human-robot safety have mainly focused on single safety components, such as collision avoidance [1], [2], [3] or collision injury minimization [4], [5], [6], but the role of each component is unclear until it is evaluated within a complete safety system. Recently, vision-only HRI safety systems have gained momentum in the industrial context [7], [8], [9]. In this work, we propose a HRI safety model for collaborative environments, specifically targeted for industrial manufacturing. For the model, we adopt the concept of workspace safety zones by Bdiwi et al. [7], [10] and we model them through captured point clouds (Figure 1). For interaction, we adopt the projector-mirror system in Vogel et al. [11], [12].

The novel contributions in this work are:

- A complete HRI safety model for collaborative manufacturing in a shared workspace S. The model is based on three zones (human, robot and danger) adapted from [7], [10] (Figure 1).
- Algorithms to detect danger zone violations and to detect and update changes in the shared workspace automatically by the robot or manually through human verification.
- An experimental setup for HRI assembly task with Cranfield benchmark parts where the proposed and a baseline safety model are experimentally evaluated.

In the experimental part, we provide quantitative results for the proposed system and a baseline system.

[1]Laboratory of Signal Processing, [2]Laboratory of Mechanical Engineering and Industrial Systems and [3]Laboratory of Automation and Hydraulic Engineering, Tampere University of Technology (TUT), Finland
First.Family@tut.fi

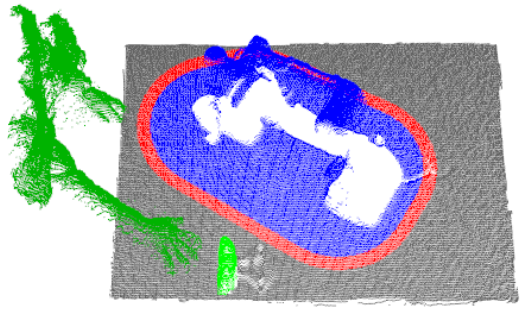


Fig. 1. Illustration of our safety model. The shared human-robot collaboration workspace S is modelled as a point cloud captured by a depth sensor. There are two zones where robot and human can freely operate, the *robot zone* Z_r (blue) and the *human zone* Z_h, respectively. The two zones are separate by the third "border zone", *danger zone* Z_d (red), where any change causes immediate halt of the robot. During normal operation changes in Z_h are recorded as change regions R_i (green) and the robot cannot move to these regions before they are verified by a human co-worker.

II. PROPOSED FRAMEWORK AND METHODS

Our shared workspace model consists of three spatial zones: robot zone Z_r, danger zone Z_d and human zone Z_h, and methods to update and monitor the zones online.

A. Depth-based workspace model

We model the workspace S as a single $W \times H$ depth map image I_S. For convenience, the depth map coordinate frame is aligned with the robot coordinate frame. Therefore the depth map is an image that directly represents the 3D structure of a monitored workspace captured from the top. In our setting, the three zones (Figure 1) represent volume and volume changes (depth) are detected.

Changes in the workspace are detected by fast and simple element-wise subtraction:

$$I_\Delta = ||I_S - I|| \tag{1}$$

where I is the most recent depth data from the depth sensor re-projected to the robot frame and sampled in a regular grid

of the size $W \times H$ that matches the current workspace model I_S. The bins (pixels) in the difference image are thresholded by a depth threshold τ ($\tau = 10mm$ used in our experiments) that detect bins where substantial changes have occurred. Operation on the detected bins depends on each zone:

$$\forall \boldsymbol{x} \;\mid\; I_\Delta(\boldsymbol{x}) \geq \tau \begin{cases} \text{if } \boldsymbol{x} \in Z_d & \textit{HALT} \\ \text{if } \boldsymbol{x} \in Z_r & I_S(\boldsymbol{x}) = I(\boldsymbol{x}) \\ \text{if } \boldsymbol{x} \in Z_h & M_h = \mathbf{0},\ M_h(\boldsymbol{x}) = 1 \end{cases} . \quad (2)$$

a) Case 1: the change has occurred in the danger zone Z_d and therefore the robot must be immediately halted to avoid collision. For maximum safety this processing stage must be executed first and must test all pixels $\boldsymbol{x}$ before the next stages.

b) Case 2: the change has occurred in the robot working zone Z_r and is therefore caused by the robot itself by moving and/or manipulating objects and therefore the workspace model I_S can be safely and automatically updated.

c) Case 3: the change has occurred in the human safety zone Z_h and we create the mask M_h that represents the changed bins. The mask is re-computed for every depth frame to allow temporal changes. Robot can continue operation normally, but if the danger zone intersects with any changed bin in M_h, then the robot is halted and changed regions must be verified by a human. Again, no automation is adopted for maximum safety. The verified regions are added to the workspace model and operation continues.

In the shared workspace, the danger zone Z_d isolates a human co-worker working in the human zone Z_h and the robot operating in the robot zone Z_r by a spatial margin ω ($\omega = 20mm$ in the experiments) which size depends on the system reaction speed or the safety regulations in the industrial standard (ISO/TS 15066).

B. Coordinate transforms

The depth map model I_S of the workspace S in Section II-A is defined in a Cartesian coordinate frame that is aligned with the robot coordinate frame and is thus our *world frame*. A 3D point $\boldsymbol{p}$ in the world frame can be transformed to the depth sensor frame by 3D rotation and translation ($\boldsymbol{p}$ in homogeneous coordinates):

$$\boldsymbol{p}' = \mathcal{T}_s^w \boldsymbol{p} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{p} \ . \quad (3)$$

In our experimental platform $\boldsymbol{R}$ and $\boldsymbol{t}$ were solved by standard calibration procedure using a checkerboard pattern. The calibration procedure also provided the intrinsic camera matrix $\boldsymbol{K}$ for the depth sensor. For simplified computation we omitted the small skew value s and the lens correction step and adopted the inverse mapping of the standard pinhole camera model and adapted it for the depth measurements of current capture $I = \{\boldsymbol{p'}\}_i = \{x, y, d\}_i$ (a depth image). Now, the point $\boldsymbol{p}$ in the world frame can be computed from (homogeneous coordinates):

$$\boldsymbol{p} = \mathcal{N}^{-1}\left(\boldsymbol{R}^T \boldsymbol{K}^{-1} \boldsymbol{p}' + \boldsymbol{t}\right) \quad (4)$$

where $\boldsymbol{K}$ is the depth camera intrinsics matrix

$$\boldsymbol{K} = \begin{bmatrix} f_x & 0 & 0 & o_x \\ 0 & f_y & 0 & o_y \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (5)$$

and $\mathcal{N}^{-1}$ is the inverse coordinate normalization function

$$\boldsymbol{p} = \mathcal{N}^{-1}(\boldsymbol{p}) = \begin{bmatrix} p_x p_z \\ p_y p_z \\ p_z \end{bmatrix} \quad (6)$$

where p_z is the depth value measured by the depth sensor.

By similar procedure we also calibrated the projector to the world (robot) frame using inverse camera calibration [13]. With the calibrated projector we can render images based on the bin locations of the model I_S.

C. Robot zone construction

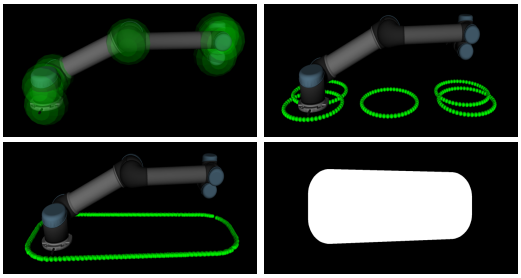


Fig. 2. Illustration of computation of the robot zone Z_r. Top-left: the selected 3D control points $\boldsymbol{\phi}_i$, i.e. robot joint locations, are provided by the robot controller (green spheres). Top-right: the control point x, y-coordinates map directly to the robot frame xy-plane - these are converted to regions by plotting circles of radius r. Bottom-left: [14] algorithm provides a convex hull around the circles. Bottom-right: convex hull is converted to a binary mask M_r that is mapped to the workspace image I_S dimensions.

Our zone modeling engine uses the 3D robot control points $\boldsymbol{\phi}_i$ for simulations (Figure 2). The control points can be read from the UR5 robot controller with the rate of 125 Hz. The robot joint locations provide sufficient information to cover all robot dimensions. Since the robot coordinate frame is also our world frame the points $\boldsymbol{\phi}_i$ can be directly mapped to the depth map I_S. In particular, we use only the x,y-coordinates that share the two axes of I_S, respectively. Around each coordinate pair we define a circle of radius r ($r = 250mm$ used in our experiments). Then we run the efficient convex hull algorithm of Graham et al. [14] that produces the actual robot zone Z_r in the same frame with I_S and can be efficiently rendered by a binary mask representation (Figure 2). All other zones can be generated from the known workspace S and the robot zone Z_r as illustrated in Figure 1.

III. EXPERIMENTS

A. A benchmarking task and evaluation

For comparing the proposed and the baseline model (Section III-C) we defined a suitable task to experiment safety in

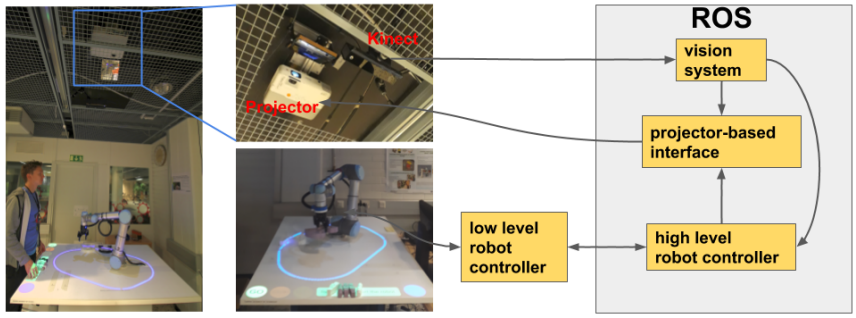


Fig. 3. Overall description of our task setup. A projector (Epson EB-905) with a mirror and a depth sensor (Kinect V2) are installed to the ceiling and pointing downwards toward the workspace. The workspace contains a flat surface (note that our model is not restricted to a flat surface) and a robotic arm (UR5) with Rototiq 85 gripper. Projector, robot and depth sensor are all connected to a single laptop computer that runs the Robot Operating System (ROS) on Ubuntu 16.04. and performs all computing. Video is available at https://youtu.be/YLF_oVbImPM.

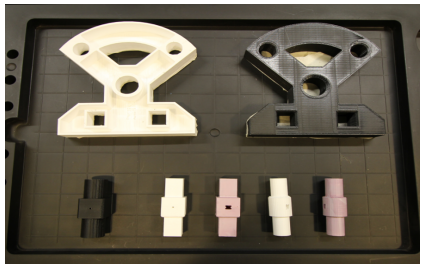


Fig. 4. Cranfield benchmark [15] parts selected for our evaluation: the front and back plates (top row) and the five "screws" that connect the plates (bottom row).

a shared workspace and to provide quantitative measures for comparisons. For this purpose we adopted the well-known *Cranfield benchmark* [15] that was designed for an assembly task. We selected 7 parts (Figure 4) and defined which assembly stages are made by a robot and which by a human. The task was the same in all experiments and consisted of the following steps:

1) A human operator enters to the workspace and starts the experiment by pressing a "START" button.
2) The robot brings the back plate to the shared workspace and goes back to collect the front plate. The human co-worker starts inserting the screws into the back plate.
3) *Proposed* – Any new item (including the screws) entering the workspace creates an "unverified region" R_i which needs to be explicitly verified by pressing the "CONFIRM". If Z_d and R_i intersect, the robot stops. *Baseline* – The robot is halted using a "STOP" button or by violating the safety line before inserting the screws.
4) If necessary the robot is restarted and the robot inserts the front plate and the task is finished.

For task performance evaluation we selected two different metrics: the *total assembly completion time* and the *robot idle time*. The total assembly time measures how long it takes for a co-worker to finish the experiment. The idle time measures how long during the experiment the robot was doing nothing.

B. Setup and configuration

The overall description of our benchmark system is shown in Fig. 3 which includes the workspace, hardware components and the main software interfaces. The workspace was captured by the depth sensor which was installed at the ceiling perpendicular to the workspace and overseeing both the robot and a human co-worker.

A standard 3LCD projector was installed to the ceiling and used to display the user interface and operational information. The projector outputs a 1920×1080 color projection image with 50 Hz frame rate. Due to the short distance from the ceiling to the workspace we increased the physical projection size by installing a mirror in 45° angle to re-project the image to the workspace. Interaction with the UI components was provided with the depth sensor (setting a hand on an UI component is detected as a depth change).

The *ur_modern_driver* ROS package [16] was used to establish a ROS interface between the high and low level robot controllers. The package provides official drivers for the Universal Robot family and two different controller modes: velocity and position (used in the experiments) based control.

C. Baseline method

For model comparison we implemented a baseline method inspired by [11], [12] using an RGB camera and a projector. The method does not specifically model or update the workspace but assumes it is a planar table. The danger zone boundaries are projected to the workspace and safety violation is detected by comparing the projected boundary to a simulated boundary. The method is based on two different masks: *current-state mask* (measured by the RGB camera) and *expected-state mask* (simulated using robot movement and known workspace structure). The boundary

was simplified by using a single line (see video[1]) without compromising the method performance.

D. Results

TABLE I
AVERAGE TASK COMPLETION TIMES AND AVERAGE ROBOT IDLE TIMES AND THEIR STANDARD DEVIATIONS FOR THE COLLABORATIVE CRANFIELD BENCHMARK ($N = 21$).

	Tot. time [s]	Robot idle [s]
Baseline	46.82 ± 6.22	13.45 ± 6.21
Our	$\mathbf{40.98 \pm 4.26}$	$\mathbf{7.66 \pm 4.28}$
Improv.	12.4%	43.5%

Human-robot interaction experiments were conducted using 21 undergraduate students of mechanical engineering and automation with various backgrounds, but nobody had prior knowledge about the system or the task. Before recorded experiments each student was introduced to the task and the user interface and they were able to test the system. The order of the experiments with our or the baseline model was chosen randomly and the robot motion (speed) was identical in both experiments. Each participant completed the task twice with both models and the runs with the smallest idle times were included to our comparison.

The results of the experiments are shown in Table I. From these results it is obvious that our model of a dynamically updated and shared workspace achieves substantial improvement even in the simple assembly task. The total improvement is 12.4% in overall performance, but the difference is particularly evident in robot idle time where improvement was more than 40%. The main reason for the improvements is the fact that the shared dynamic workspace in our model allows parallel working in the same shared workspace. It is important to notice that our system does not compromise anything in safety but provides more flexible working than the baseline. The baseline method does not allow simultaneous operation since it cannot update the workspace model and therefore any changes are detected as violations.

IV. CONCLUSIONS

We proposed a HRI safety system for collaborative manufacturing that requires only a single depth sensor and a single projector. The proposed system provides a flexible workspace modelling where human and robot can alter workspace contents (manipulate new or existing objects) and still avoid working too close to each other (defined by the danger zone width). We conducted real experiments on a simple assembly task where the proposed model provided clear improvements as compared to a baseline method without depth sensing or the dynamic workspace model with safety zones.

[1] https://youtu.be/YLF_oVbImPM

ACKNOWLEDGMENT

This work was supported by the UNITY project funded by Teknologiateollisuuden 100-vuotissäätio and Jane and Aatos Erkko Foundation, and the Academy of Finland project: "Competitive funding to strengthen university research profiles", decision number 310325.

REFERENCES

[1] L. Wang, B. Schmidt, and A. Y. Nee, "Vision-guided active collision avoidance for human-robot collaborations," *Manufacturing Letters*, vol. 1, no. 1, pp. 5–8, 2013.

[2] M. Saveriano and D. Lee, "Distance based dynamical system modulation for reactive avoidance of moving obstacles," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5618–5623, IEEE, 2014.

[3] A. Mohammed, B. Schmidt, and L. Wang, "Active collision avoidance for human–robot collaboration driven by vision sensors," *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 9, pp. 970–980, 2017.

[4] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3356–3363, IEEE, 2008.

[5] A. De Luca and F. Flacco, "Integrated control for phri: Collision avoidance, detection, reaction and collaboration," in *IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pp. 288–295, IEEE, 2012.

[6] A. Cirillo, F. Ficuciello, C. Natale, S. Pirozzi, and L. Villani, "A conformable force/tactile skin for physical human–robot interaction," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 41–48, 2016.

[7] M. Bdiwi, "Integrated sensors system for human safety during cooperating with industrial robots for handing-over and assembling tasks," *Procedia CIRP*, vol. 23, pp. 65–70, 2014.

[8] C. Morato, K. N. Kaipa, B. Zhao, and S. K. Gupta, "Toward safe human robot collaboration by using multiple kinects based real-time human tracking," *Journal of Computing and Information Science in Engineering*, vol. 14, no. 1, p. 011006, 2014.

[9] C. Vogel, C. Walter, and N. Elkmann, "Safeguarding and supporting future human-robot cooperative manufacturing processes by a projection-and camera-based technology," *Procedia Manufacturing*, vol. 11, pp. 39–46, 2017.

[10] M. Bdiwi, M. Pfeifer, and A. Sterzing, "A new strategy for ensuring human safety during various levels of interaction with industrial robots," *CIRP Annals*, vol. 66, no. 1, pp. 453–456, 2017.

[11] C. Vogel, M. Poggendorf, C. Walter, and N. Elkmann, "Towards safe physical human-robot collaboration: A projection-based safety system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3355–3360, IEEE, 2011.

[12] C. Vogel, C. Walter, and N. Elkmann, "A projection-based sensor system for safe physical human-robot collaboration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5359–5364, IEEE, 2013.

[13] I. Martynov, J.-K. Kamarainen, and L. Lensu, "Projector calibration by "inverse camera calibration"," in *Scandinavian Conference on Image Analysis (SCIA)*, 2011.

[14] R. L. Graham and F. F. Yao, "Finding the convex hull of a simple polygon," *Journal of Algorithms*, vol. 4, no. 4, pp. 324–331, 1983.

[15] K. Collins, A. Palmer, and K. Rathmill, *Robot Technology and Applications*, ch. The Development of a European Benchmark for the Comparison of Assembly Robot Programming Systems. 1985.

[16] T. T. Andersen, "Optimizing the universal robots ros driver.," tech. rep., Technical University of Denmark, Department of Electrical Engineering, 2015.

PUBLICATION
IV

Proof of concept of a projection-based safety system for human-robot collaborative engine assembly

A. Hietanen, A. Changizi, M. Lanz, J.-K. Kämäräinen, P. Ganguly, R. Pieters and J. Latokartano

*International Conference on Robot and Human Interactive Communication*2019, 1–7

Proof of concept of a projection-based safety system for human-robot collaborative engine assembly

Antti Hietanen[1], Alireza Changizi[1], Minna Lanz[1], Joni Kämäräinen[2], Pallab Ganguly[1], Roel Pieters[1] and Jyrki Latokartano[1]

***Abstract*— In the past years human-robot collaboration has gained interest among industry and production environments. While there is interest towards the topic, there is a lack of industrially relevant cases utilizing novel methods and technologies. The feasibility of the implementation, worker safety and production efficiency are the key questions in the field. The aim of the proposed work is to provide a conceptual safety system for context-dependent, multi-modal communication in human-robot collaborative assembly, which will contribute to safety and efficiency of the collaboration. The approach we propose offers an addition to traditional interfaces like push buttons installed at fixed locations. We demonstrate an approach and corresponding technical implementation of the system with projected safety zones based on the dynamically updated depth map and a graphical user interface (GUI). The proposed interaction is a simplified two-way communication between human and the robot to allow both parties to notify each other, and for the human to coordinate the operations.**

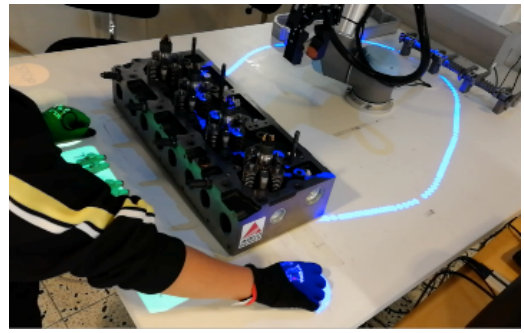


Fig. 1. Collaborative human-robot assembly with projection-based safety system.

I. INTRODUCTION

Within all areas of robotics, the demand for collaborative and more flexible robot systems is expected to rise [1], [2]. In the automation industry, for example, industrial and collaborative robotics are acting as a driver for market growth. The past decade has therefore seen a growing interest in this technology, for an economic relevance of bringing humans and robots closer together in the manufacturing working environment [3], [4]. The continued rise of industrial robots certainly seems to be inevitable, being driven by a variety of production demands, including the need for safer and more simplified robotic technologies to work in collaboration with humans, increased resource efficiency, and continued adaptation to the proliferation of automation and the Internet of Things (IoT) [5].

In recent years, research on human-robot interaction (HRI) and human-robot collaboration (HRC) has also increased [6], [7]. Manufacturing companies have gradually increased the implementation of collaborative robots assisted by machine vision into their daily production [8], [9]. Flexibility and changeability of assembly processes must be increased, therefore, more advanced interaction and/or collaboration between the operator and the assembly system is required (see Fig. 1). Such interaction is expected to improve complex assembly processes by increasing flexibility of the total system [10]. Particularly in physical interaction, where a worker guides a robot or the robot provides power assistance to the worker, the expectations are high. Furthermore, as concluded in [11] and by our own work [4] it is expected that in the future semi-automated robotized assembly requires industrial robots to collaborate with human workers as part of a team to complete tasks based on their individual competences.

One way to improve the effectiveness of human-robot collaboration is supporting human workers with current task- and context-dependent work instructions via suitable communication modalities. This implies that information presented to an operator should only be relevant to the current task and not to future tasks. Similarly, presented information should be in useful form by utilizing appropriate modalities [12]. For example, a safety zone should be projected around and on top of a shared work space, such that it is most visible for the operator. This approach aims to raise the human understanding of the task in the context of safe human-robot collaboration. The other way around, workers should be supported in controlling the robot or other components of the production system by using suitable modalities as well. Traditional interfaces, such as push buttons, might be less suitable to halt robot motion than, for example, the crossing of virtual and projected light barriers [13], [14]. The developments proposed in this work address these particular issues (see Fig. 1).

In detail, this paper introduces a safety system for context-dependent, multi-modal communication in a collaborative assembly task and presents its implementation towards a real industrial mid-heavy (< 5 kg) assembly task. These developments are an extension of our previous developments [15], which described a prototype of the vision- and projector-based safety system. In particular, the contributions of this

[1] Automation Technology and Mechanical Engineering, [2] Computing Sciences, Tampere University of Technology, Finland first.last@tuni.fi

work are therefore:

1) Development of a user interface and interaction to enable suitable task and context-dependent safety communication between human and robot (i.e. projection of current safety zone, virtual buttons).
2) Development of a safety-zone monitoring system that reacts when the safety zone is violated.
3) Implementation of a case study based on a real industrial assembly task.
4) Implementation of a task sequencing and work allocation schedule between human and robot resources for the industrial assembly task.

This paper is organized as follows. Section II provides the theoretical background on the topics of human-machine interaction and human-robot collaboration. Section III proposes the technical developments of the safety system. Section IV presents the industrial case study. Finally, Section V reports conclusions and future works.

II. Theoretical Background

A. Human Machine Interaction

Human-machine interfaces allow interaction between humans and machines. For reasons of safety and ergonomics, it is crucial that the design of such interaction is smooth, productive and has taken into account the persons comfort and well being. Human-factors which must be considered when designing interaction can be categorized in either physical ergonomics or mental ergonomics (Machine directives regarding ergonomics; Directive 2006/42/EC, Annex I, 1.1.6 "Ergonomics", EN 614 (parts 1-2) Safety of machinery - Ergonomic design principles, CEN, Brussels).

Standards (EN 13861) must be followed which state that the machinery should be designed in the context and consistent with human capabilities, limitations and needs. Therefore, analysis has to be carried out that assesses the effect of the interaction design on the persons safety, health and well-being. Safety must be ensured by considering these characteristics with respect to dangerous areas in a machine. If necessary, extra safety systems have to be installed that halt the machine when a certain barrier is crossed (e.g. light barrier, fence).

Health and well being for physical ergonomics is related to repetitive and awkward body movements or heavy loads. Ergonomics principles for mental workload relate to a person's cognitive abilities and its immediate effects due to the interactive scenario (ISO 10075 series (parts 1-3), Ergonomic principles related to mental workload). Safety aspects due to reduced cognitive abilities may result to long-term health effects and a higher risk of accidents. Cognitive abilities such as attention, memory, reasoning and perception can therefore have a large effect on the safety of interaction. The design of information exchange for interaction is, however, not as simple as reducing mental workload. Mental fatigue, experienced by either too much (repetitive) information or too little (monotonous) information, can have equal effects. Additionally, allocation of tasks between machines and robots is quite challenging and depends largely on the task and the human factors involved in the collaborative setup [16], [17].

B. Human-Robot Collaboration

Safety for humans in collaboration actions can be ensured by different strategies and methods. The ISO 10218-1/ 2 (2011) standards give safety requirements for robots and robot systems in industrial context, where collaborative operation between a person and a robot sharing a common work space is taken into account. Technical specification (TS) 15066 (2016) provides additional guidance for safe HRI. The standards define four collaborative safeguarding modes:

1) Safety-rated monitored stop: The robot cannot enter the collaborative workspace when a human is present in the workspace. When the the robot is present in the workspace and the human enters, the safety-rated monitored stop is activated.
2) Hand guiding: Hand-guided motion of the robot is allowed, when a human is guiding the robot with a hand-guiding tool. Again, when a person enters the workspace in which a robot is present, the safety-rated monitored stop is activated.
3) Speed and separation monitoring: Safety is guaranteed by ensuring a minimum separation distance between a human and the robot. The separation distance depends on the speed of the robot i.e. lower robot speed allows a smaller separation distance. When the separation distance goes below the protective separation distance, the robot is halted.
4) Power and force limiting: Physical contact between a human and the robot is allowed. Risk reduction is done by keeping robot-related hazards below certain limits that are defined in the risk assessment.

There exist numerous studies regarding safety in human-robot interaction and collaboration that adhere to, and even surpass, the standard safeguarding modes. The most relevant works related to our approach are described as follows: Matthias [3] proposed a safety concept with seven levels. Relevant risks can be assessed and reduced to harmless levels by applying suitable measures. The risk/impact evaluation was divided into six levels and give a solid background for system design and risk management. Marvel [18] proposed a set of metrics to evaluate speed and separation monitoring efficiency in shared, industrial work spaces. More recently, Marvel and Norcross [19] proposed an approach for implementing speed and separation monitoring in collaborative robot work cells. Lasota [7] divided safe human-robot interaction into four methods: safety through control, safety through motion planning, safety through prediction and safety through consideration of psychological factors. Pre- and post-collision control methods are included in safety through control. In pre-collision, safety is ensured by using methods such as safety regions, tracking separation distance, and guiding robot motion away from humans. In post-collision, approaches intend to minimize injuries after the

detection of a collision. The aim of safety through motion planning is to compute robot paths and motions that avoid collisions. Lasota [7] pointed out that psychological safety should be taken into consideration by adjusting robot behavior specifically towards the human comfort of interaction.

Additionally, there exist several different types of vision-based safety, monitoring and guidance systems in the field of robotics, however, these are mostly in research state and not commercially available. Halme et al. [4] made an extensive literature review covering recent developments from the field. One notable exception is the Pilz Safety EYE that is available for commercial use [20]. Particularly, Vogel et al. [21] developed a safety monitoring system that uses one or multiple cameras and user interaction provided by a projector. The coexistence of a human and a robot considers hybrid cells classified into the shared tasks and workspace. In such case, the tasks and workspaces of the human and the robot are sequential and are shared accordingly. The design of the HRC cell currently asks for considerable time as there are no ready-made guidelines for such design [22], [23], [24].

III. Shared Workspace Model

The aim of this work is to provide a conceptual safety system for context-dependent, multi-modal communication in human-robot collaborative assembly. To achieve this, a real industrial, manual assembly task was taken and redefined towards a human-robot collaborative assembly task. The computer vision- and projection-based safety system monitors the workspace and enables communication and interaction for the shared assembly task. The depth map of the workspace is continuously updated, and this information is used to ensure the safety of the human.

A. Depth-based workspace model

The workspace S is modelled as a single W $\times$ H depth map image I_S and it represents the geometric structure of the workspace. The model can be updated during run time automatically by the robot itself or by a human co-worker using the proposed user interface described in Section IV-C. The workspace model and the virtual robot zones (Sec. III-B) are aligned with the real robot using a calibrated geometric transformation between the depth sensor and the robot. The workspace is in 3D space (a point cloud), however, since occluded regions could not be monitored with a single depth sensor and due to computational complexity, we found the 2D representation more suitable. A 2D map representation of the workspace allows for fast operations (collision detection, map updating) during run-time compared to other representations such as point clouds or voxel grids.

B. Robot zone concept

On the shared workspace model (collaborative workspace) three spatial zones are generated: robot zone Z_r, danger zone Z_d and human zone Z_h, which are illustrated in Fig. 2. The zones Z_r and Z_h represent spaces where the robot and the human can operate freely, respectively. The robot zone Z_r is initialized using set of control points C_r containing minimum

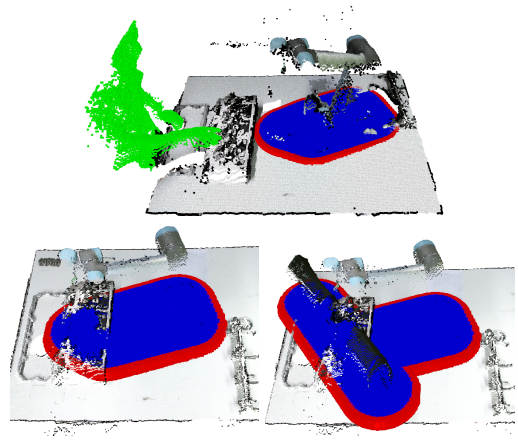


Fig. 2. A shared workspace S is modelled as a depth map image I_S aligned with the robot coordinate system. The robot zone Z_r (blue) is dynamically updated and subtracted from I_S to generate the human zone Z_h (gray). The two zones are separated by the danger zone Z_d (red) which is monitored for safety violations. Changes in Z_h are recorded to binary masks R_i (green). Manipulated objects are enclosed by Z_d and automatically added to Z_r (bottom right).

number of 3D points covering all the extreme parts of the robot. The point locations in the robot frame are calculated online using a modified version of the Hawkinses model [25] and projected to I_S. Finally, the projected points are converted to regions having radius of ω and a convex hull [26] enclosing all the regions is computed and the resulting hull is rendered as a binary mask M_r representing Z_r.

The human and robot zone are separated by the danger zone Z_d, where any change causes immediate halt of the robot. The danger zone is constructed by adding a danger zone margin $\Delta\omega$ and then subtracting Z_r from the results:

$$Z_d = M_r(\omega + \Delta\omega) \setminus Z_r. \tag{1}$$

Conceptually, the 2D danger zone Z_d is related to the protective separation distance concept used in the technical standards, which captures a scalar distance value.

The human zone mask Z_h is easy to compute as a binary operation since the human zone is all pixels not occupied by the robot or danger zone:

$$Z_h = I_S \setminus (Z_r \cup Z_d). \tag{2}$$

Workspace changes are detected by monitoring the difference between the current depth data I from the depth sensor and the current workspace model I_S:

$$I\Delta = \|I_S - I\| \tag{3}$$

Substantial changes on $I\Delta$ are detected using a threshold value τ and the operation on the detected bins depends on which zone they lie in:

$$\forall \boldsymbol{x} \;\mid\; I_{\Delta}(\boldsymbol{x}) \geq \tau \begin{cases} \text{if } \boldsymbol{x} \in Z_d & \textit{HALT} \\ \text{if } \boldsymbol{x} \in Z_r & I_S(\boldsymbol{x}) = I(\boldsymbol{x}) \\ \text{if } \boldsymbol{x} \in Z_h & M_h = 0,\ M_h(\boldsymbol{x}) = 1 \end{cases}, \quad (4)$$

where M_h is a 2D binary mask containing clustered anomalies R_i. During the run time the algorithm must first iterate over all the pixels from the $I\Delta$ image and check if any of them fall inside the danger zone Z_d before doing any further processing. If a pixel is detected inside the Z_d the robot is immediately halted and the robot must be manually reset to continue. After the first condition check all the pixels are evaluated against the robot working zone Z_r. If a change has occurred inside Z_r then the workspace model I_S is automatically updated. The main purpose of the zone is to register the movements and/or object manipulations of the robot itself and safely updated them to the model.

Finally, if the pixel has passed all the other checks, the change has occurred in the human safety zone Z_h and we create the mask M_h to represent the changed bins. Note that the mask is re-created for every measurement to allow for temporal changes, however, this does not affect robot operation. The robot continues operation normally, but if its danger zone intersects with any changed bin in M_h, the robot is halted. The changed bins must be verified manually by the human co-worker via our graphical user interface rendered by a projector. If the bins are verified, then these values are updated to the workspace model I_S and operation continues normally. The robot and the human co-worker are isolated to their own operational spaces Z_r and Z_h, respectively, using the danger zone Z_d which spatial margin size is configurable and depends on the system computing - "reaction" -speed. The size of the margin was selected so that the robot had enough time to stop before any parts of the new obstacle could enter inside Z_r. In this work, the margin was heuristically, set to 50 mm. The workspace and safety model is explained in details in [15], where a benchmark experiment is used to demonstrate its capabilities.

C. Ensuring safety during object manipulation

During a collaborative task the robot can carry sharp or heavy object (up to 5kg with UR5) which can potentially harm the human co-worker. In this work we propose an important extension to our previous work [15] by extending the robot zone when the robot is carrying task related objects which exceed the default robot zone. In this scenario we use the known geometric properties of the task related object and the robots grasp point to add new control points to the kinematic model of the robot online. Finally the binary mask M_{obj} for the object is created similarly as M_r and the final shape of the zones are computed by fast binary operations:

$$Z_r = M_r(\omega) \cup M_{obj}(\omega), \quad (5)$$

$$Z_d = M_r(\omega + \Delta\omega) \cup M_{obj}(\omega + \Delta\omega) \setminus Z_r \ . \quad (6)$$

In this work ω and $\Delta\omega$ are fixed but can be easily updated during run-time for instance based on the extended protective distance definition (ISO TS 15066).

IV. Industrial Case study

The research and developments are motivated by a real industrial assembly case taken from a local diesel engine manufacturing company (see Fig. 5). The task is the sub-assembly of mid-heavy parts ($<$ 5 kg) to an engine block, as this allowed collaborative robots to be considered. Such collaborative robots have suitable workspace, object handling properties (payload), and offer programmable interaction (hand-guiding). Currently, the studied case is executed via manual assembly, which resulted in a bottleneck in the manufacturing line. Considering a collaborative robot in assembly has the benefit of parallel assembly and the utilization of both the robots and operators expertise (e.g. robot strength, human compliance to tasks), while not introducing the complexities of complete automation. The study considers the feasibility of the implementation with projector and computer vision-based safety system and user experience in the task. Detailed description of the evaluation and results from quantitative and qualitative evaluations with respect to performance, safety and ergonomy are presented in the follow-up paper [27]. Details of the case study, its setup and user interaction are described as follows.

A. Case study description

To evaluate our proposed safety system the manual assembly task was redefined where a human and a robot can work safely in a collaborative manner (see Fig. 3). The safety through consideration of psychological factors [7] was considered in such manner that robot movements were slow, stopping distances at comfortable level, and approach angles were in the field of view of the user. The task consists of a tractor diesel engine sub-assembly that, in the current assembly line, is handled manually. The sub-assembly tasks we consider are the installation of the eight rocker arms, the motor frame, the rocker shaft and finally inserting and tightening the bolts to secure the parts (see Fig. 5). The choice of which steps should be handled by the robot, the operator or as a collaborative assembly depends on many factors, such as object properties (e.g. weight, size, grasp capabilities), assembly complexity (accuracy, speed, robustness) and capabilities of both operator and robot. In this case, the robot handled the heavy parts (frame and rocker shaft) and the operator handled the lighter parts and the assembly (rocker arms, nuts and bolts). The tasks that robot and human are doing, are both parallel and sequential depending on the part they handle (Fig. 4).

Placing the rocker arms and the operations related to bolts and nuts are done individually by the human worker whereas the fetching and positioning of motor frame is done by the robot, further illustrated in Fig. 3. Placing the rocker shaft onto the motor is a delicate task and requires precise positioning, thus this step is done using physical hand-guidance. At first, the robot fetches the rocker shaft

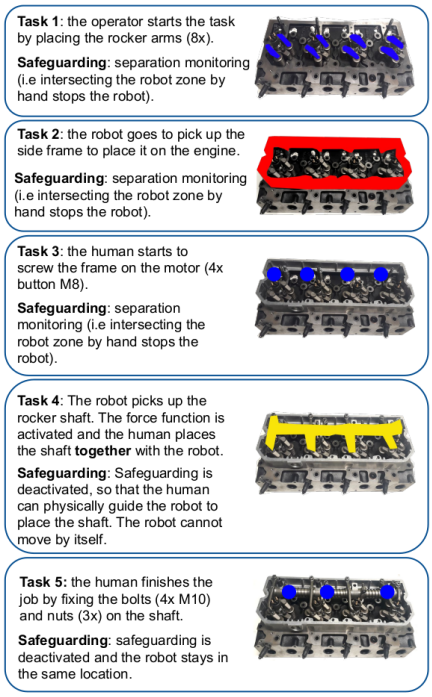


Fig. 3. Description of the assembly steps and resource allocation between human and the robot. The five tasks are conducted by the operator (blue) or the robot (red) or both (yellow)

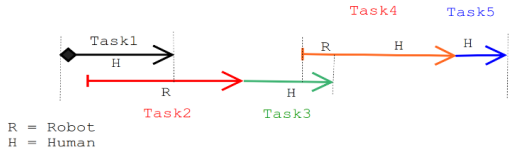


Fig. 4. Task sequencing and work allocation between the resources

and brings it above the engine. After the shaft has arrived and the robot has completely stopped, the safety system is deactivated and the robot is automatically set to a force mode where external forces can be used to move the robot tool center point (TCP) location. The mode requires axis specific forces to be defined which the robot is to apply to its environment. The amount of force to apply is dependent on the weight of the object the robot is carrying and the muscular strength of the human worker. For instance setting the forces too small the TCP will start to drift gradually along the incorrectly set force axis. However, setting the forces too large, the human worker might not have enough strength to drag or pull the TCP along different axes. In the experiments, for all the participants the forces were set to -30N, 10N and 60N along x, y and z-axis respectively.

The physical ergonomics of the assembly task were considered by having the heavy objects being lifted and guided by the robot. Hand-guiding of the robot with a grasped object is physically ergonomic to the person due to the compensation of the weight of the object and the compliant motion of the robot. Additionally, no buttons need to be pressed to proceed in the sequence of the assembly task. Interaction for potential object or safety zone violations is done via virtual buttons that are projected on the surface of the work environment. Mental stress reduction is considered via appropriate configuration of robot behavior (robot stops when the operator crosses the safety zone) and via the communication of safety zones (projection of safety zone on the table). A video recording of a complete experiment can be found here: `https://youtu.be/v-hM4Nycua4`.

B. Setup and Configuration

Our benchmark system is depicted in Fig. 6. This includes the workspace, hardware components and the main software interfaces. At the top of the workspace a depth sensor (Kinect V2) monitors the area and sends information about the state of the workspace for the robot and the projector. A projector (Epson EB-905) is used to display UI components and robot safety boundaries. The workspace is captured by the depth sensor which is installed at the ceiling perpendicular to the workspace and overseeing both the robot and a human co-worker. The depth sensor works at 30 Hz and provides 512×424 size depth map giving a spatial depth resolution of approximately 5.4 mm $\times$ 5.4 mm on the table surface (2 meters from camera). Due to noisy sensor measurement the resolution was reduced to 10 mm $\times$ 10 mm. The IAI Kinect2 library [28] was used to capture, process and deliver depth images to ROS.

A standard 3LCD projector was installed to the ceiling and used to display the user interface and operational information. The projector outputs a 1920×1080 color projection image with 50 Hz frame rate. The physical projection area is increased by installing a mirror in 45° angle to re-project the image to the workspace. The depth sensor and the projector were aligned with the robot's base frame using a standard chessboard calibration method. The ROS interface for the UR5 drivers (communication between high and low-level robot controllers) was taken from the UR modern driver ROS package [29].

C. User interface and interaction

The robot safety zone and UI components were reflected on the planar surface via the projector, and user interaction with the UI was enabled by detecting a change in depth of each individual UI component. The created user interface is shown in Fig. 5.

The START and STOP buttons were the main UI components of the interface model that enable robot operation or stop it immediately, respectively. During initial testing of the system it was noted that the participants could easily start the robot accidentally by bending over the GO button, thus an ENABLE button (colored in blue) was added to the other side of the UI space which needs to be pressed

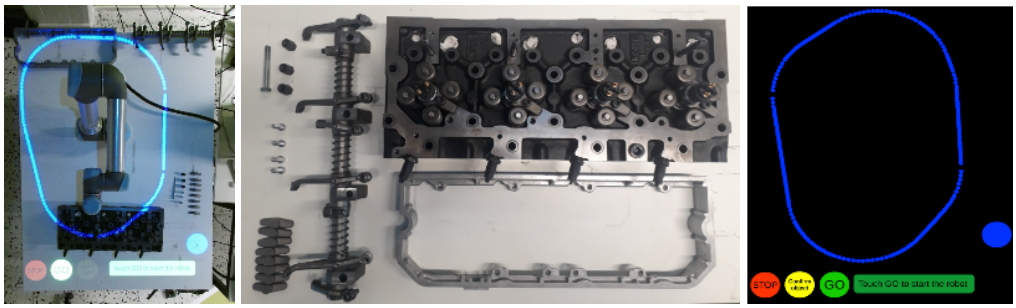


Fig. 5. Left: Top view of the projected work area with the robot and the engine block, surrounded by the assembly parts. Middle: engine block (right top), surrounded by the engine frame (bottom right), rocker shaft (left), rocker arms (bottom left), nuts and bolts. Right: user interface illustrated with buttons, an info bar and a current form of the safety zone.

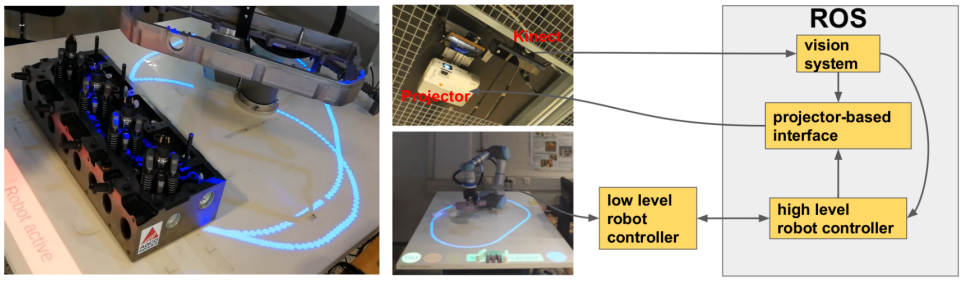


Fig. 6. Overview of the system used in the experiment. Video of a complete experiment: `https://youtu.be/v-hM4Nycua4`

simultaneously with the START button. To manually update the workspace model representation described in Section III another CONFIRM button was added to allow the human operator to confirm unverified regions from the workspace that can be safely added to the workspace model. CONFIRM also requires simultaneous activation of the ENABLE button. In addition to the interaction UI buttons, robot intentions as well as instructions related to the task are displayed for the human operator using an INFORMATION BAR component.

D. Technical limitations

During the experiments it was noted that the average delay between a pause command send by the high-level robot controller until the robot had completely stopped was measured 100 ms. In addition, the projector had another latency of 58 ms in rendering, but this could be avoided with a better projector. Due to the latencies we limited the robot maximum speed to 50% of the maximum to achieve the safe stopping distance (determined heuristically). The average inference time of the algorithm described in Section III with 250×250 size workspace depth map I_S over all user studies was 30 ms. The main computational bottleneck of our method is the heavy preprocessing of the input depth map, which is required due to the noise measurements of the Kinect sensor. Specifically, objects having very reflective or dark surfaces as well as the areas close to depth discontinuities are problematic and result in corrupted depth estimates and missing information as shown in the point cloud in Fig. 7.

The update and detection rules in Eq. 4 are sensitive to noise and to suppress the effect of these noise pixels we adopt the Euclidean clustering proposed by [30]. The method decomposes the 3D points x, y and Δz of ΔI into clusters based on their Euclidean distance and filters out small sparse clusters. This step is essential for robustness of our method but requires extra computation that depends on the number of points in the ΔI image. However, this process step can be made faster by just down-sampling the depth map. In the experiments the algorithm was configured to filter out clusters having less than 200 points which corresponded roughly an object having 1 cm radius in real world on our setup. The current interaction and visualization components assume a known static surface (flat table). In a dynamic workspace where robot intentions and interaction components have to be precisely projected a more robust tracking-and-projection system has to be implemented [29].

V. Conclusions

In this paper we proposed a method to support the two-way interaction with a robot by using a vision and a projection-based modality. The approach is described from a technical and user's perspective and can offer a seamless, human-centered and safe collaboration in mid-heavy ($<$ 5 kg) assembly operations. The approach for realizing the GUI

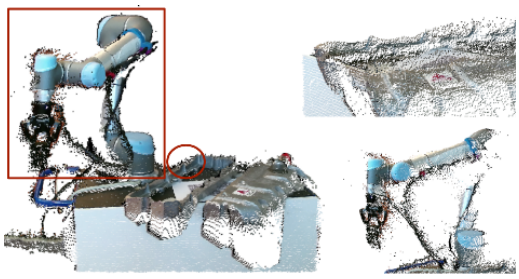


Fig. 7. Sparse outliers (flying pixels) are visible near object boundaries and dark surfaces (below right) and object deformed due to reflective material.

offers an alternative to traditional interfaces like push buttons installed at fixed locations. Experiments were conducted in a laboratory setting with an industrial product (diesel engine assembly). A preliminary user experience assessment indicated that the projector based visual guidance system and GUI, as well as the developed safety system were either considered user friendly, comfortable and safe. The user experience is formally assessed in the follow-up work [27]. The future work includes experiments with more traditional industrial robots, which are not designed as inherently safe. Future work will also further develop and test this approach in a real industrial environment.

ACKNOWLEDGMENT

This work was supported by the UNITY project funded by Teknologiateollisuuden 100-vuotisstio and Jane and Aatos Erkko Foundation; the Academy of Finland project: Competitive funding to strengthen university research pro- files, decision number 310325; and the European Union's Horizon 2020 research and innovation programme under grant agreement No 825196.

REFERENCES

[1] I. F. of Robotics, "World robotics 2017 edition," 2017, https://ifr.org/free-downloads/, Last accessed on 2018-05-04.

[2] Eurobotics, "Robotics PPP roadmap," 2017, https://ec.europa.eu/research/industrial_technologies/pdf/robotics-ppp-roadmap_en.pdf, Last accessed on 2018-02-20.

[3] B. Matthias, S. Kock, H. Jerregard, M. Kallman, I. Lundberg, and R. Mellander, "Safety of collaborative industrial robots: Certification possibilities for a collaborative assembly robot concept," in *ISAM*. IEEE, 2011, pp. 1–6.

[4] R.-J. Halme, M. Lanz, J. Kämäräinen, R. Pieters, J. Latokartano, and A. Hietanen, "Review of vision-based safety systems for human-robot collaboration," *Procedia CIRP*, vol. 72, no. 1, pp. 111–116, 2018.

[5] Techemergence, "Global competition rises for ai in-dustrial robotics," 2017, https://www.techemergence.com/global-competition-rises-ai-industrial-robotics/, Last accessed on 2018-5-8.

[6] B. Chandrasekaran and J. M. Conrad, "Human-robot collaboration: A survey," in *SoutheastCon 2015*. IEEE, 2015, pp. 1–8.

[7] P. A. Lasota, T. Fong, J. A. Shah, *et al.*, "A survey of methods for safe human-robot interaction," *Foundations and Trends® in Robotics*, vol. 5, no. 4, pp. 261–349, 2017.

[8] A. Cherubini, R. Passama, A. Crosnier, A. Lasnier, and P. Fraisse, "Collaborative manufacturing with physical human–robot interaction," *RCIM*, vol. 40, pp. 1–13, 2016.

[9] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding, and B. Matthias, "Safety in human-robot collaborative manufacturing environments: Metrics and control," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 882–893, 2016.

[10] J. Krüger, V. Katschinski, D. Surdilovic, and G. Schreck, "Flexible assembly systems through workplace-sharing and time-sharing human-machine cooperation (pisa)," in *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*. VDE, 2010, pp. 1–5.

[11] R. Ahmad and P. Plapper, "Human-robot collaboration: Twofold strategy algorithm to avoid collisions using tof sensor," *International Journal of Materials, Mechanics and Manufacturing*, vol. 4, no. 2, pp. 144–147, 2015.

[12] C. Kardos, Z. Kemény, A. Kovács, B. Pataki, and J. Váncza, "Context-dependent multimodal communication in human-robot collaboration," *PROCEDIA CIRP*, vol. 72, pp. 15–20, 2018.

[13] N. Nikolakis, V. Maratos, and S. Makris, "A cyber physical system (cps) approach for safe human-robot collaboration in a shared workplace," *RCIM*, vol. 56, pp. 233–243, 2019.

[14] G. Tang, P. Webb, and J. Thrower, "The development and evaluation of robot light skin: A novel robot signalling system to improve communication in industrial human–robot collaboration," *RCIM*, vol. 56, pp. 85–94, 2019.

[15] A. Hietanen, R.-J. Halme, J. Latokartano, R. Pieters, M. Lanz, and J.-K. Kämäräinen, "Depth-sensor-projector safety model for human-robot collaboration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Robotic Co-workers 4.0*, 2018.

[16] G. Salvendy, *Handbook of human factors and ergonomics*. John Wiley & Sons, 2012.

[17] M. S. Sanders and E. J. McCormick, *Human factors in engineering and design*. McGRAW-HILL book company, 1987.

[18] J. A. Marvel, "Performance metrics of speed and separation monitoring in shared workspaces," *IEEE Transactions on automation Science and Engineering*, vol. 10, no. 2, pp. 405–414, 2013.

[19] J. A. Marvel and R. Norcross, "Implementing speed and separation monitoring in collaborative robot workcells," *RCIM*, vol. 44, pp. 144–155, 2017.

[20] Pilz, "Safetyeye," 2018, https://www.pilz.com/en-INT/eshop/00106002207042/SafetyEYE-Safe-camera-system, Last accessed on 2018-15-11.

[21] C. Vogel, C. Walter, and N. Elkmann, "Safeguarding and supporting future human-robot cooperative manufacturing processes by a projection-and camera-based technology," *Procedia Manufacturing*, vol. 11, pp. 39–46, 2017.

[22] M. P. Mayer, B. Odenthal, M. Faber, C. Winkelholz, and C. M. Schlick, "Cognitive engineering of automated assembly processes," *Human factors and ergonomics in manufacturing & service industries*, vol. 24, no. 3, pp. 348–368, 2014.

[23] P. Tsarouchi, G. Michalos, S. Makris, T. Athanasatos, K. Dimoulas, and G. Chryssolouris, "On a human–robot workplace design and task allocation system," *IJCIM*, vol. 30, no. 12, pp. 1272–1279, 2017.

[24] I. Aaltonen, T. Salmi, and I. Marstio, "Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry," *Procedia CIRP*, vol. 72, pp. 93–98, 2018.

[25] K. P. Hawkins, "Analytic inverse kinematics for the universal robots ur-5/ur-10 arms," Georgia Institute of Technology, Tech. Rep., 2013.

[26] R. L. Graham and F. F. Yao, "Finding the convex hull of a simple polygon," *Journal of Algorithms*, vol. 4, no. 4, pp. 324–331, 1983.

[27] A. Hietanen, R. Pieters, M. Lanz, J. Latokartano, and J.-K. Kämäräinen, "Ar-based interaction for human-robot collaborative manufacturing," *RCIM*, to appear.

[28] T. Wiedemeyer, "Iai kinect2: Tools for using the kinect one (kinect v2) in ros," 2015.

[29] R. S. Andersen, O. Madsen, T. B. Moeslund, and H. B. Amor, "Projecting robot intentions into human environments," in *2016 25th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2016, pp. 294–301.

[30] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, 2010.

PUBLICATION
V

AR-based interaction for human-robot collaborative manufacturing
A. Hietanen, R. Pieters, M. Lanz, J. Latokartano and J.-K. Kämäräinen

Robotics and Computer-Integrated Manufacturing 63.(2020)

Robotics and Computer Integrated Manufacturing 63 (2020) 101891

Contents lists available at ScienceDirect

Robotics and Computer Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Full length Article

AR-based interaction for human-robot collaborative manufacturing

Antti Hietanen, Roel Pieters, Minna Lanz*, Jyrki Latokartano, Joni-Kristian Kämäräinen

Tampere University, Korkeakoulunkatu 6, Tampere, Finland

ARTICLE INFO

Keywords:
Human-robot collaboration
Assembly
Augmented reality
User studies

ABSTRACT

Industrial standards define safety requirements for Human-Robot Collaboration (HRC) in industrial manufacturing. The standards particularly require real-time monitoring and securing of the minimum protective distance between a robot and an operator. This paper proposes a depth-sensor based model for workspace monitoring and an interactive Augmented Reality (AR) User Interface (UI) for safe HRC. The AR UI is implemented on two different hardware: a projector-mirror setup and a wearable AR gear (HoloLens). The workspace model and UIs are evaluated in a realistic diesel engine assembly task. The AR-based interactive UIs provide 21–24% and 57–64% reduction in the task completion and robot idle time, respectively, as compared to a baseline without interaction and workspace sharing. However, user experience assessment reveal that HoloLens based AR is not yet suitable for industrial manufacturing while the projector-mirror setup shows clear improvements in safety and work ergonomics.

1. Introduction

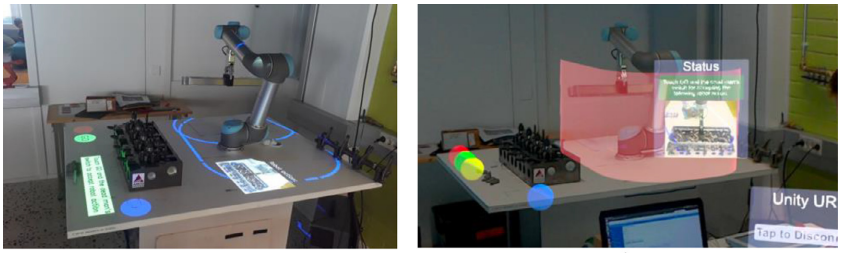
In order to stay competitive, European small and medium-sized enterprises (SMEs) need to embrace flexible automation and robotics, information and communications technologies (ICT) and security to maintain efficiency, flexibility and quality of production in highly volatile environments [1]. Raising the output and efficiency of SMEs will have a significant impact on Europe's manufacturing and employment capacity. Robots are no longer stand-alone systems in the factory floor. Within all areas of robotics, the demand for collaborative and more flexible systems is rising as well [2]. The level of desired collaboration and increased flexibility will only be reached if the systems are developed as a whole including perception, reasoning and physical manipulation. Industrial manufacturing is going through a process of change toward flexible and intelligent manufacturing, the so-called Industry 4.0. Human-robot collaboration (HRC) will have a more prevalent role and this evolution means breaking with the established safety procedures as the separation of workspaces between robot and human operator is removed. However, this will require special care for human safety as the existing industrial standards and practices are based on the principle that operator and robot workspaces are separated and violations between them are monitored.

HRC has been active in the past to realize the future manufacturing expectations and made possible by several research results obtained during the past five to ten years within the robotics and automation scientific communities [3]. In particular, this has involved novel mechanical designs of lightweight manipulators, such as the Universal Robot family and KUKA LBR iiwa. Due to the lightweight structure, slow speed, internal safety functions and impact detection, the robots are considered a more safe solution for close proximity work than traditional industrial robots. The collaborative robots can be inherently safe, but the robotic task can create safety hazards for instance by including sharp or heavy objects that are carried at high speed. In order to guarantee the safety of the human co-worker, a large variety of external multi-modal sensors (camera, laser, structured light etc.) has been introduced and used in robotics applications to prevent collisions [4,5]. In order to transfer research solutions from the lab to industrial settings they need to comply with strict safety standards. The International Organization for Standardization (ISO) Technical Specification (TS) 15066 [6] addresses in detail the safety with industrial collaborative robotics and defines further four different collaborative scenarios. The first specifies the need and required performance for a safety-rated, monitored stop (robot moving is prevented without an emergency stop conforming to the standard). The second outlines the behaviors expected for hand-guiding a robot's motions via an analog button cell attached to the robot. The third specifies the minimum protective distance between a robot and an operator in the collaborative workspace, below which a safety-rated, controlled stop is issued. The fourth limits the momentum of a robot such that contact with an operator will not result in pain or injury.

The main focus of this work is to define a model to monitor safety margins with a depth sensor and to communicate the margins to the

* Corresponding author.
E-mail address: minna.lanz@tuni.fi (M. Lanz).

https://doi.org/10.1016/j.rcim.2019.101891
Received 2 November 2019; Accepted 2 November 2019
Available online 21 November 2019



a b

Fig. 1. The proposed interactive UIs for safe human-robot manufacturing: a) projector-mirror and b) wearable (AR) HoloLens. Video: https://youtu.be/-WW0a-LEGLM.

operator with an interactive User Interface (UI), illustrated in Fig. 1. The work focuses on the third scenario of ISO/TS where the operator-robot distance is communicated interactively.

This paper proposes a shared workspace model for HRC manufacturing and interactive UIs. The model is based on the virtual zones introduced by Bdiwi et al. [7]: robot zone and human zone. In the human zone an operator can freely move and the robot is not allowed to enter. The robot zone is dynamically changing based on robot tasks and if the operator or any other object enters the robot zone, the robot is halted. In the proposed model, the two zones are separated by a safety monitored danger zone and any changes in the workspace model, either from the robot or operator side, cause halting the robot. The purpose of the safety zone is to allow dynamic update of the workspace model without compromising safety. The proposed workspace model, safety monitoring and UIs in the work are consistent with their collaboration levels Level 1 and Level 2 proposed by Bdiwi et al. [7]. The work belongs to the Safety Through Control category. Instead of a passive system this paper proposes a safety model which allows a dynamic AR-based interaction for HRC.

The paper is organized as follows. First, Section 2 describes briefly the background for safe HRC in industrial settings and reviews the current state-of-the-art. Section 3 explains the proposed shared workspace model in detail and in Section 4 two different AR-based UIs integrated to the proposed model are discussed. Next, Section 5 explains the experimental setup for evaluating the workspace model and UIs in a realistic assembly task. Finally, in Section 6 the results from the experiments are reported and conclusions are drawn in Section 7.

2. Related work

2.1. Human-robot collaboration in manufacturing

HRC in manufacturing context aims at creating work environments where humans can work side-by-side with robots in close proximity. In such setup, the main goal is to achieve efficient and high-quality manufacturing processes by combining the best of both worlds: strength, endurance, repeatability and accuracy of robots complemented by the intuition, flexibility and versatile problem solving skills of humans. During a collaboration task, the first priority is to ensure safety of the human co-worker. Vision sensors have been a popular choice to gain information from the surrounding environment, which is crucial for safe trajectory planning and collision avoidance. Other sensing modalities, such as pressure/force, can be combined with visual information to enhance the local safety sensing [8]. In addition to the safety aspect, one of the key challenges in industrial HRC is the interaction and communication between the human and robot resources [9]. According to Liu and Wang [10] the ICT system should be able to provide information feedback and support a worker in the HRC manufacturing. In industrial settings, the physical environment (i.e. floor, tables) can be used as a medium where task-related information, such as boundaries of the safe working area or user interface components can be projected.

In the literature, several recent works have demonstrated their HRC systems on real industrial manufacturing tasks, where both aspects, safety and communication, are considered. Vogel et al. [11] presented a collaborative screwing application where a projector-camera based system was used to prevent collision and display interaction and safety-related information during the task. In [12] the authors proposed a wearable AR-based interface integrated to an off-the-shelf safety system. The wearable AR supports the operator on the assembly line, by providing virtual instructions on how to execute the current task in the form of textual information or 3D model representation of the parts. The integrated interface in [12] was utilized in an automotive assembly task where a wheel group was installed as a shared task. De Gea Fernández [13] and Magrini [14] fused sensor data from different sources (IMU, RGB-D and laser) and a standardized control and communication architecture was used for safety robot control. Human actions and intentions were recognized through hand gestures and the systems were validated in a real industrial task from the automotive industry. While the mentioned implementations are good examples of safe HRC in manufacturing, the works are mainly technological demonstrations and do not provide data from qualitative or quantitative evaluations that could further emphasize the need of HRC. More similar to this work, a context-aware mixed reality approach was utilized in car door assembly and evaluated against two baseline methods (printed and screen display instructions) [15]. From the experiments, quantitative (efficiency and effectiveness of the task completion) as well as qualitative data (human-robot fluency, trust in robot etc.) were measured through recordings and questionnaires, respectively.

2.2. Safety standards, guidelines and strategies

The manufacturing industry leans on industrial standards that define safety requirements for HRC and, therefore, it is important to reflect research to the existing standards. One of the first attempts to define the work guidelines between human and robot was the ISO 10218–1/2 [16, 17] standards, describing the safety requirements for robot manufacturers and robot system integrators. However, the safety requirements were not comprehensively discussed as the current Industry 4.0 requires more flexible HRC. TS 15066 [6] was introduced to augment the existing standards and for instance added a completely new guideline for the maximum biomedical limits for different human body parts in HRC. The ISO/TS combination defines four techniques for collaborative operation for collaborative applications: *safety-rated monitored stop (SMS), hand-guiding operation (HG), speed and separation monitoring (SSM)* and *power and force liming (PFL)*.

Recently, several authors have provided design guidelines and concepts corresponding to next-generation manufacturing and aligned with today's safety standards. Marvel [18] proposed a set of metrics to evaluate SSM efficiently in shared workspaces. In contrast, Sloth et al.

[19] estimated the highest velocity a collaborative robot arm can reach, while still complying with PFL. Bdiwi et al. [7] proposed four different levels of interaction in HRC. In the bottom level, the robot and human work inside the same working space but have separate tasks. In the other end, the human and robot have a shared task with physical interaction. In each level, different types of safety functions are developed, linked and analyzed. In this paper the described taxonomies are used as a guideline, defining the safety requirements and standards for the implemented HRC application. In [20, 21], the safety issue was discussed from the perspective when the collision between the human and robot cannot be necessarily avoided. The authors summarized three different strategies for safety: *crash safety* (controlled collision using power/force control), *active safety* (external sensors for collision prediction) and *adaptive safety* (applying corrective actions that lead to collision avoidance). Lasota et al. [22] provided a comprehensive survey of existing safety strategies in HRC and divided the methods into four different directions: *Safety Through Control, Safety Through Motion Planning, Safety Through Prediction* and *Safety Through Consideration of Psychological Factors*.

2.3. Vision-based safety systems

Safety through control is the most active research field in HRC safety, where the collision is prevented for instance by stopping or slowing down the robot through the use of methods including defining safety regions or tracking separation distance [4]. One of the earliest approaches in industrial environments is to use volumetric virtual zones, where a movement inside a certain zone would signal an emergency stop or slowing down the robot. SafetyEYE (Pilz) [23] and SafeMove (ABB) [24] are few standardized and commercialized vision-based safety systems that use an external tracking system to monitor movement inside predefined safety regions. Similar to the proposed safety system in this paper, the authors [25,26] presented an approach where the regions can be updated during run-time. In [26] a dynamic robot working area is projected on a flat table by a standard digital light processing (DLP) projector and safety violations are detected by multiple RGB cameras that inspect geometric distortions of the projected line due to depth changes. Moreover, recent research [27–29] have discussed an efficient and probabilistic implementation of SSM as dictated by the ISO 15066, where the safety system has dynamic control of the safety distance between the robot and human operator such that it complies with the minimum safety requirements.

Depth sensing has become a popular and efficient approach to monitor the shared environment and to prevent collision between the robot and an unknown object (e.g., a human operator). In most of the approaches a virtual 3D model of the robot is generated and tracked during run-time while real measurements of the human operator from the depth sensor are used to calculate the distance between robot and human body parts. Depth sensing is then combined with reactive and safety-oriented motion planning that guides the manipulator to prevent collisions [30–32]. For a practical application these methods have to be extended to multi-sensor systems where the possibility of having occluded points is removed [33]. Current consumer-grade RGB-D sensors can deliver up to several million point measurements in a second which requires substantial computational power. For real-time interaction more complex implementations have been proposed such as GPU-based processing [34] and efficient data-structures [35]. In contrast, this work combines depth sensing with zone-based separation monitoring (see Section 3), ensuring safe interaction without an expensive feature tracking system and complex implementation of real time motion planning. In [36] a vision-based neural network monitoring system is proposed for locating the human operator and ensuring a minimum safety distance between the co-workers. In parallel, deep models have been proposed for human hand and body posture recognition [37] and intention recognition in manufacturing tasks [38]. However, most of the learning-based approaches assume all human actions to be from a known observation set and are not designed to work for unseen actions, making them less practical for complex tasks.

2.4. AR-based operator support systems

Advances in display and vision technologies have created new interaction modalities that enable informative and real-time communication in shared workspaces. In robotics, various different signaling techniques have been proposed during the years and one common way is to project 2D information to table or floor [39]. One of the earliest approaches to create a communication interface between robot and human was introduced in [40]. The paper presents a system that visually tracks the operator's pointing hand and projects a mark at the indicated position using an LCD projector. The marker is then utilized by the robot in a pick-and-place task. More recently, Vogel et al. [11] used a projector to create a 2D display with virtual interaction buttons and textual description that allow intuitive communication. In another recent work [15,41] the authors proposed a projector-based display for HRC in industrial car door assembly. In contrast to other projector-based works, the system can display visual cues on complex surfaces. User studies of the systems against two baselines, a monitor display and simple text descriptions, showed clear improvements in terms of effectiveness and user satisfaction. Wearable AR such as head-mounted displays (HMD) and stereoscopic glasses have recently gained momentum as well. Earliest versions of wearable AR devices were typically considered bulky and ergonomically uncomfortable when used over long periods of time [42]. In addition, each of the human participants in the collaborative task is required to wear the physical device. However, 2D displays can only provide limited expression power and can be more easily interfered, for instance, due to direct sunlight or obstructing obstacles. In [43] a HMD was used for robot motion intent communication, which evaluated the method's effectiveness against a 2D display in a simple toy task. Huy et al. [44] demonstrated the use of HMD in an outdoor mobile application where a projector system cannot be used. Elsdon and Demiris [45] introduced a handheld spray robot where the control of the spraying was shared between human and robot. In [12] the authors combined two wearable AR-gear, a head-mounted display and a smartwatch, for supporting operators in shared industrial workplaces.

While the advances of AR technologies have increased their usage in HRC applications, it is unclear how mature the wearable AR gear technology is for real industrial manufacturing. Therefore, this paper investigates HRC safety with two different AR-based UIs, wearable AR and projector-based AR, that are evaluated in a real diesel engine assembly task. The UIs are used together with the proposed safety system that establishes dynamic collaborative zones as defined in Bdiwi [7]. The shared workspace is then modelled and monitored using a single depth sensor installed on the ceiling overseeing all actions in the workspace.

3. The shared workspace model

In the model, a shared workspace S is modelled with a single depth map image I_s and divided to three virtual zones: robot zone Z_r, human zone Z_h and danger zone Z_d (Fig. 2). The zones are modelled by binary masks in the same space as I_s which makes their update, display and monitoring fast and simple. The depth map image I_s is aligned with the robot coordinate system. The robot zone Z_r (blue) is dynamically updated and subtracted from I_s to generate the human zone Z_h (gray). The two zones are separated by the danger zone Z_d (red) which is monitored for safety violations. Changes in Z_h are recorded to binary masks M_i (green). Manipulated objects are automatically added to Z_r, see Fig. 2c.

3.1. Depth-based workspace model

The work considers a shared workspace monitored by a depth

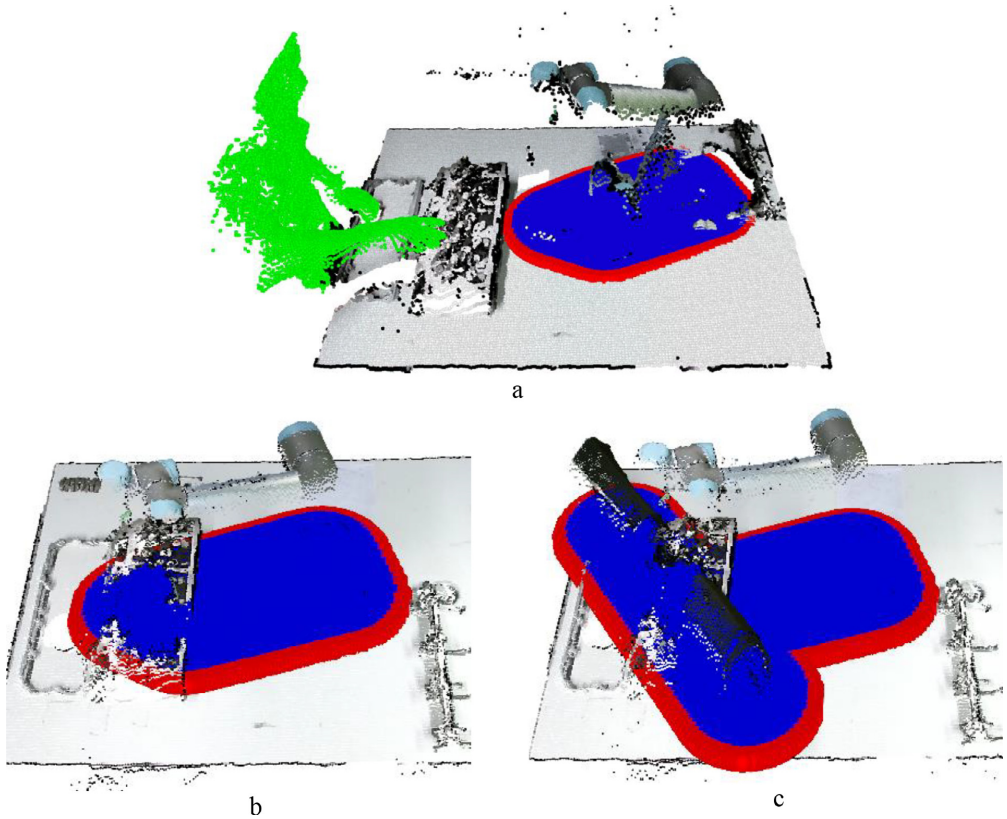


Fig. 2. a) Shared workspace S is modelled as a depth map image where three virtual zones are defined: robot zone (blue), human zone (gray) and danger zone (red); b) robot approaching to grasp an object; c) robot zone extended to cover the carried object.

sensor which can be modelled as a pin-hole camera parametrized by two matrices: the intrinsic camera matrix K, modelling the projection of a Cartesian point to an image plane, and the extrinsic camera matrix (R|t), describing the pose of the camera in the world. The matrices can be solved by the chessboard calibration procedure [46]. For simplicity the model uses the robot coordinate frame as the world frame.

After calibration, the points p in the depth sensor plane can be transformed to a Cartesian point in the world frame and finally to the workspace model $I_s = \{x\}_i$ of the size $W \times H$:

$$P = N^{-1}(RK^{-1}p + t) \tag{1}$$

$$x = T_{proj}P \tag{2}$$

where N^{-1} is the inverse coordinate transformation and T_{proj} is the projective transformation. Now, computations are done efficiently in I_s and (1) is used to display the results to the AR hardware and (2) to map the robot control points (Section 3.2) to the workspace model.

3.2. Binary zone masks

Since all computation is done in the depth image space I_s the three virtual zones can be defined as binary masks of the size $W \times H$: the robot zone Z_r, the danger zone Z_d and the human zone Z_h.

a) *The robot zone mask Z_r:* The zone is initialized using set of control points C_r containing minimum number of 3D points covering all the extreme parts of the robot. The point locations in the robot frame are calculated online using a modified version of the robot kinematic model and projected to I_S. Finally, the projected points are converted to regions having radius of ω and a convex hull [47] enclosing all the regions is computed and the resulting hull is rendered as a binary mask M_r representing Z_r.

b) *The danger zone mask Z_d:* Contour of the Z_r and constructed by adding a danger margin $\Delta\omega$ to the robot zone mask and then subtracting Z_r from the results:

$$Z_d = M_r(\omega + \Delta\omega) \setminus Z_r \tag{3}$$

c) *The human zone mask Z_h:* This is straightforward to compute as a binary operation since the human zone is all pixels not occupied by the robot zone Z_r or the danger zone Z_d:

$$Z_h = I_s \setminus (Z_r \cup Z_d) \tag{4}$$

3.3. Adding the manipulated object to Z_r and Z_d

An important extension of the model is that the known objects that the robot manipulates are added to the robot zone Z_r and Z_d (see Fig. 2c). This guarantees that the robot does not accidentally hit the operator with an object it is carrying. In such case a new set of control points C_{obj} is created using known dimensions of the object and the robot current configuration. Finally, the binary mask M_{obj} for the object

is created similarly as M_r and the final shape of the zones are computed by fast binary operations:

$$Z_r = M_r(\omega) \cup M_{obj}(\omega) \tag{5}$$

$$Z_d = M_r(\omega + \Delta\omega) \cup M_{obj}(\omega + \Delta\omega) \setminus Z_r \tag{6}$$

3.4. Safety monitoring

The main safety principle is that the depth values in the danger region Z_d must match with the stored depth model. Any change must produce immediate halt of the system. The depth-based model in the robot frame I_s provides now fast computation since the change detection is computed as a fast subtraction operation

$$I_\Delta = \|I_s - I\| \tag{7}$$

where I is the most recent depth data transferred to same space as the workspace model. The difference bins (pixels) are further processed by Euclidean clustering [48] to remove spurious bins due to noisy sensor measurements. Finally, the safety operation depends on which zone a change is detected:

$$\forall\, x \mid I_\Delta(x) \geq \tau \begin{cases} if\ x \in Z_d\ (HALT) \\ if\ x \in Z_r\ \ I_s(x) = I(x) \\ if\ x \in Z_h\ M_h = 0,\ \ M_h(x) = 1 \end{cases} \tag{8}$$

where τ is the depth threshold. In the first case, the change has occurred in the danger zone Z_d and therefore the robot must be immediately halted to avoid collision. For maximum safety this processing stage must be executed first and must test all pixels x before the next stages.

In the second case, the change has occurred in the robot working zone Z_r and is therefore caused by the robot itself by moving and/or manipulating objects and therefore the workspace model I_s can be safely updated. In the last case, the change has occurred in the human safety zone Z_h and therefore the mask M_h is created that represents the changed bins (note that the mask is recreated for every measurement to allow temporal changes, but it does not affect robot operation). Robot can continue operation normally, but if its danger zone intersects with any 1-bin in M_h, then these locations must be verified from the human co-worker via the proposed UIs.

If the bins are verified, then these values are updated to the workspace model I_s and operation continues normally. Note that the system does not verify each bin separately, but a spatially connected region of changed bins. This operation allows a shared workspace and arbitrary changes in the workspace which do occur away from the danger zone.

4. The user interfaces

The danger zone defined in Section 3.2 and various UI components are rendered to graphical objects in two AR setups, shown in Fig. 3.

4.1. UI components

The proposed UI contains the following interaction components (Fig. 3): 1) a danger zone that shows the region operators should avoid; 2) highlighting changed regions in the human zone; 3) *GO* and *STOP* buttons to start and stop the robot; 4) *CONFIRM* button to verify and add changed regions to the current model; 5) *ENABLE* button that needs to be pressed simultaneously with the *GO* and *CONFIRM* buttons to take effect; and 6) a graphical display box (image and text) to show the robot status and instructions to the operator.

The above UI components were implemented to two different hardware, projector-mirror and HoloLens. The UI components and layout were the same for the both hardware to be able to compare the human experience on two different types of hardware.

4.2. Projector-mirror AR

The projector-mirror setup is adopted from [11,49,26] with the main difference that the multiple RGB cameras are replaced with a single RGB-D sensor (Kinect v2). A standard 3LCD projector is installed to the ceiling to point to a 45° tilted mirror that re-projects the picture to the workspace area. The mirror is needed to expand the projection area of the standard projector but could be replaced with a wide-angle lens projector. The projector outputs a 1920 × 1080 color image with 50 Hz frame rate. The projector coordinate frame is calibrated to the world (robot) coordinate frame using the inverse camera calibration with a checkerboard pattern [50].

4.3. Wearable AR (HoloLens)

As a state-of-the-art head-mounted AR display, Microsoft HoloLens is adopted. The headset can operate without any external cables and the 3D reconstruction of the environment as well as accurate 6-DoF localization of the head pose is provided by the system utilizing an internal IMU sensor, four spatial-mapping cameras, and a depth camera. The data exchange between HoloLens and the proposed model is done using wireless TCP/IP. For the work a Linux server was implemented that synchronizes data from the robot simulator (ROS) to HoloLens and back. As HoloLens is not a safety rated equipment at this development phase, the safety and monitoring system is used, but not shown to the user. In Fig. 4 is illustrates the working posture with HoloLens.

The interaction buttons are displayed as semi-transparent spheres that are positioned similar to the projector-mirror UI (Fig. 1). In addition, the safety region is rendered as a solid virtual fence. The fence is rendered as a polygonal mesh having semi-transparent red texture. From the 2D boundary and a fixed fence height the fence mesh is constructed from rectangular quadrilaterals that are further divided to two triangles for the HoloLens rendering software.

The UI component and the virtual fence coordinates P are defined in the robot frame and transformed to the HoloLens frame by

$$P' = (T_{AR}^{R} T_{H}^{AR})^{-1} P \tag{9}$$

where T_{AR}^{R} is a known static transformation between the robot and an AR marker (set manually to the workspace) and $T_{H}^{AR}$is the transformation between the marker and the user holographic frame. Once the pose has been initialized the marker can be removed and during run time T_{H}^{AR} is updated by HoloLens software.

5. Engine assembly task

The task used in the experiments is adopted from a local diesel engine manufacturing company. In addition to the proposed safety model and the interaction interfaces, a baseline method where the human and robot cannot work side-by-side is presented for comparison.

5.1. Task description

The task consists of five sub-tasks (Task 1–5) that are conducted by the operator (blue) or the robot (red) or both (yellow). Task 4 is the collaborative sub-task where a rocker shaft is held by the robot and carefully positioned by the operator.

The task used in the experiments is a part of a real engine assembly task from a local company. The task is particularly interesting as one of the sub-tasks is to insert a rocker shaft that weights 4.3 kg and would therefore benefit from HRC. The task is illustrated in Fig. 5 which also shows the five sub-tasks (H denotes the human operator and R the robot):

- Task 1) Install 8 rocker arms (H),
- Task 2) Install the engine frame (R),
- Task 3) Insert 4 frame screws (H),

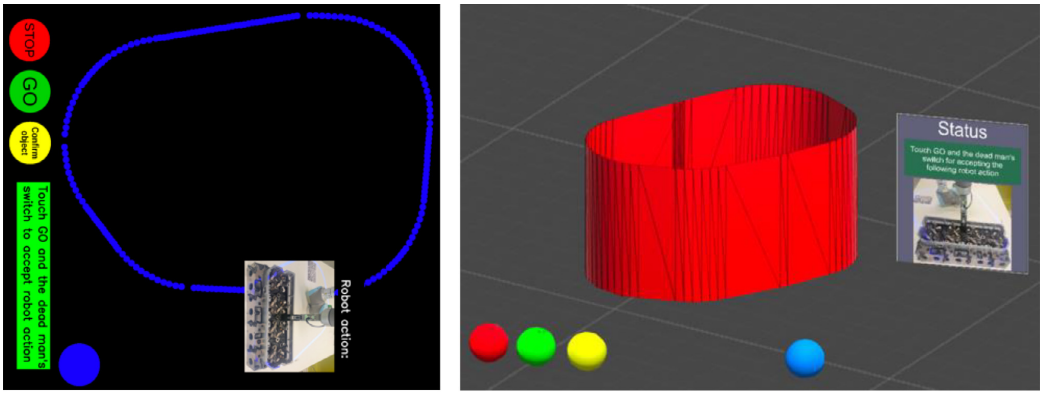


Fig. 3. UI graphics: a) projector-mirror as a 2D color image and b) the HoloLens setup rendered in Unity3D engine.

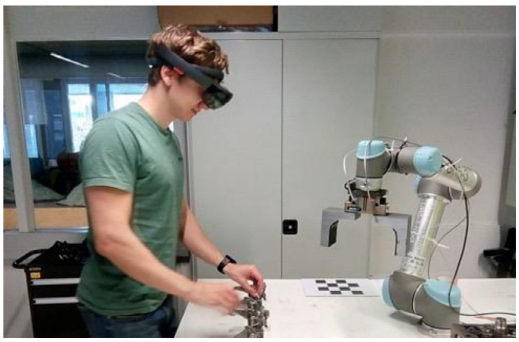


Fig. 4. Test set-up before the experiment with HoloLens.

- Task 4) Install the rocker shaft ($R + H$) and
- Task 5) Insert the nuts on the shaft (H).

Tasks 1–3 and 5 are dependent so that the previous subtask must be completed before the next can begin. Task 4 is collaborative in the sense that the robot brings the shaft and moves to a force mode allowing physical hand-guidance of the end-effector. In the force mode, the robot applies just enough force to overcome the gravitational force of the object while still allowing the human to guide the robot arm for accurate positioning.

5.2. A non-collaborative baseline

The baseline system is based on the current practices in manufacturing - the human and robot cannot operate in the same workspace simultaneously. In the setting, the operator must stay 4 m apart from the robot when the robot is moving and the operator is allowed to enter the workspace only when the robot is not moving. In this scenario the collaborative Task 4 is completely manual, the robot only brings the part. Safety in the baseline is ensured by an enabling switch button which the operator needs to press all the time for the robot to be operational. The baseline does not contain any UI components, but in the user studies the subjects are provided with textual descriptions for all sub-tasks.

6. Experiments

In this section quantitative and qualitative results are reported for the assembly task and the three different setups are compared.

6.1. Settings

The experiments were conducted using the model 5 Universal Robot Arm (UR5) and OnRobot RG2 gripper. Kinect v2 was used as the depth sensor installed to the ceiling and capturing the whole workspace area. The AR displays, the projector or HoloLens, were connected to a single laptop with Ubuntu 16.04 OS and it performed all computations. In the study, a safe work environment was implemented. The interaction is facilitated with a collaborative robot, reduced speed and force and by the projection of safety zones on the work environment. A risk

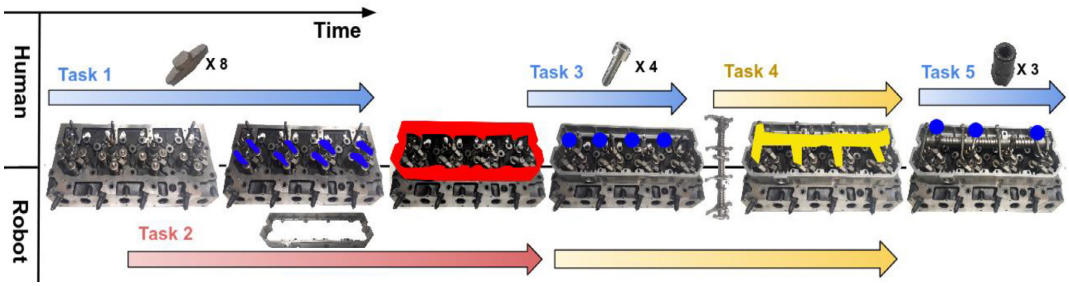


Fig. 5. The engine assembly task used in the experiments.

Table 1
The questionnaire template developed and used in the user experience studies.

Categories	Individual questions
Safety	Ql. The job has a low risk of accident.
	Q2. The safety system improves the workflow in a safe way.
InformationProcessing	Q3. A lot of time was required to learn the equipment used on the job.
	Q4. The job requires me to analyze a lot of information.
Ergonomics	Q5. Body posture and movement arrangements on the job are suitable.
	Q6. The job requires a lot of physical effort.
	Q7. The job requires a great deal of muscular strength.
Autonomy	Q8. During task, I felt a sense of choice and freedom in the things I undertake.
	Q9. Robot system considers how I would like to do things.
Competence	Q10.1 feel disappointed with my performance in my task.
	Qll. Robot conveyed confidence in my ability to do well in my task.
Relatedness	Q12.1 feel my relationship with robot at the task was just superficial.
	Q13. Robot I work with is friendly.

assessment based on [51] was carried out and residual risks were deemed acceptable.

6.2. User studies

The experiments were conducted with 20 unexperienced volunteered university students. Responsible conduct of research and procedures for handling allegations of misconduct in Finland's instructions by the Finnish Advisory Board on Research Integrity were followed. The ethics Committee of the Tampere region, hosted by University of Tampere, provides ethical guidelines for conducting non-medical research in the field of the human science. These guidelines are outlined as i) Respecting the autonomy of research subjects, ii) Avoiding harm, and iii) Privacy and data protection based on guidelines of The Finnish Advisory Board on Research Integrity. The participation was not mandatory and participants could leave any time they chose.

The data collection included collection of performance times, that were recorded, and after experimenting the three systems they were asked the questionnaire in Table 1. No personal data was collected during the experiment. The goal of the questionnaire was to evaluate physical and mental stress aspects of the human co-workers during the task. The questions were selected to cover safety, ergonomics and mental stress experience as defined in Salvendy et al. [52] and autonomy, competence, and relatedness in Deci et al. [53]. Users were asked to score each question using the scale from 1 (totally disagree) to 5 (totally agree).

6.3. Quantitative performance

For quantitative performance evaluation two different metrics were used, Average total task execution time and Average total robot idle time, that measure the total performance improvement and the time robot is waiting for the operator to complete her tasks, respectively.

The results in Fig. 6 show that the both AR-based interactive systems outperform the baseline where the robot was not moving in the same workspace with an operator. The difference can be explained by the robot idle time which is much less for AR-based interaction. The difference between the HoloLens and projector-based systems is marginal. On average, the AR-based systems were 21–24% and 57–64% faster than the baseline in the terms of the total execution time and the robot idle time respectively.

6.4. Subjective evaluation

Since the results from the previous quantitative evaluation of system performance were similar for the both HoloLens and projector-based AR interaction the user studies provided important information about the differences of the two systems.

All the 20 participants answered to the 13 template questions (Q1-Q13) listed in Table 1, and the results analyzed. The average scores with the standard deviations are shown in Fig. 7. The overall impression is that the projector-based display outperforms the two others (HoloLens and baseline), but surprisingly HoloLens is found inferior to the baseline in many safety related questions. The numerical values are given in Table 2 and these verify the overall findings. The projector-based method is considered the safest and the HoloLens-based method most unsafe with a clear margin.

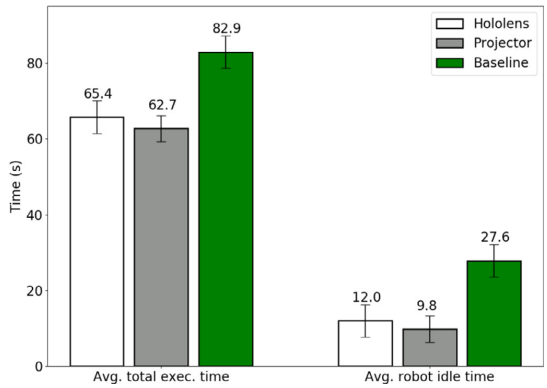


Fig. 6. Average task execution and robot idle times from the user studies.

Based on the analysis the results and free comments from the user studies, the HoloLens is experienced most unsafe due to the intrusiveness of the device. Even though it is used as augmented display (information virtually added to the scene), it blocks, to some extent, the view of the operator. Additionally, the device is quite heavy, which can create discomfort and decrease the feeling of safety. The projector-based system does not experience these features and, therefore, is experienced most safe. The amount of information needed to understand the task is smallest for the baseline while projector-based has very similar numbers and again the HoloLens-based method was found clearly more difficult to understand.

Ergonomics-wise the HoloLens and projector-based methods were superior likely to the fact that they provided help in installing the heavy rocker shaft. The autonomy numbers are similar for all methods, but the projector-based is found the easiest to work with. The users also found their performance best with the projector-based system (Competence). The question Q12 was obviously difficult to understand for the users, but all users found the system with AR interaction more plausible (Q13) than the baseline without interaction. Overall, the projector-based AR interaction in collaborative manufacturing was found safer and more ergonomic than the baseline without AR interaction and also the HoloLens-based AR.

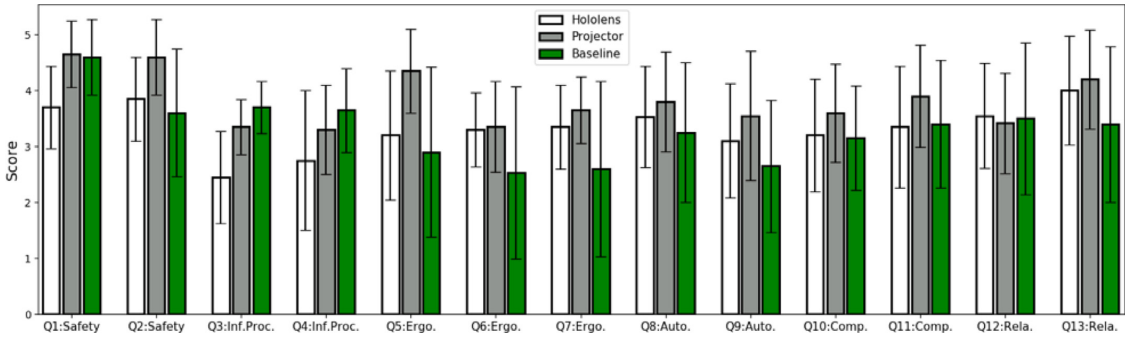


Fig. 7. Average scores for the questions Q1-Q13 used in the user studies (20 participants). Score 5 denotes "totally agree" and 1 "totally disagree" and scores for the questions Q3, Q4, Q6, Q7 and Q10 are inverted for better readability (score 5 has the same meaning as for other questions).

Table 2
Average scores for the question (Q1-Q13). Higher is better except for those marked with "¬". The best result emphasized (multiple if no statistical significance).

		HoloLens	Projector	Baseline
Safety	Ql	3.7	**4.7**	**4.6**
	Q2	3.9	**4.6**	3.6
InformationProcessing!	Q3¬	2.6	1.7	**1.3**
	Q4¬	2.3	1.7	**1.4**
Ergonomics	Q5	3.2	**4.4**	2.9
	Q6¬	**1.7**	**1.7**	2.5
	Q7¬	1.7	**1.4**	2.4
Autonomy	Q8	3.5	**3.8**	3.3
	Q9	3.1	**3.6**	2.7
Competence	Q10¬	1.8	**1.4**	1.9
	Qll	3.4	**3.9**	3.4
Relatedness	Q12	**3.6**	**3.4**	**3.5**
	Q13	**4.0**	**4.2**	3.4

Below are free comments from the user studies that well point out the reasons why different systems were preferred or considered difficult to use:

- HoloLens:
  - "Too narrow field of view, head has to be rotated a lot."
  - "Feels heavy and uncomfortable after a while."
  - "Holograms feels to be closer than they actually are."
- Projector:
  - "I would choose the projector system over HoloLens"
  - "Easier and more comfortable to use"
- Baseline:
  - "System could be fooled by placing object on the switch button."

7. Conclusions

This paper described a computation model of the shared workspace in HRC manufacturing. The model allows to monitor changes in the workspace to establish safety features. Moreover, the paper proposed a UI for HRC in industrial manufacturing and implemented it on two different hardware for AR, a projector-mirror and wearable AR gear (HoloLens). The model and UIs were experimentally evaluated on a realistic industrial assembly task and results from quantitative and qualitative evaluations with respect to performance, safety and ergonomics, and against a non-shared workspace baseline were evaluated. In experiments on a realistic assembly task adopted from the automotive sector both AR-based systems were found superior in performance to the baseline without a shared workspace. However, the users found the projector-mirror system clearly more plausible for manufacturing work than the HoloLens setup. The other AR research papers considering traditionally conveyed AR e.g. via monitors or tablets reported that AR technologies receives positive feedback from the potential users. The studies agree with this indication, except when using wearable AR such as head mounted HoloLens. The wearable AR requires still more technical maturity (in design, safety and software side) in order to be considered suitable for industrial environments. The future work includes experiments in a multimachine work environment, where the human worker operates together with more traditional industrial robots (payload up to 50 kg) and mobile robots. In addition, improvements on the existing interfaces are planned based on the feedback received form the end users, such as projecting the UI components on movable/adjustable table for increased comfort. Lastly, future experiments include the latest generation of Microsoft HMD (HoloLens 2) that has improved on the previous technical, visual, and functional aspects of HoloLens 1.

Declaration of Competing Interest

We promise no conflict of interest exits in the submission of this manuscript, and this manuscript is approved by all authors for publication.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825196.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.rcim.2019.101891.

References

[1] H. Flegel, Manufuture High-Level Group, Manufuture Vision 2030: Competitive, Sustainable And, Manufuture Implementation Support Group, 2018.
[2] euRobotics, "Robotics 2020 - strategic research agenda for robotics in Europe," 2013, p. 101.
[3] L. Wang, R. Gao, J. Váncza, J. Krüger, X. Wang, S. Makris, G. Chryssolouris, Symbiotic human-robot collaborative assembly, CIRP Ann. 68 (2) (2019) 701–726.
[4] R.-J. Halme, M. Lanz, J. Kämäräinen, R. Pieters, J. Latokartano, A. Hietanen, Review of vision-based safety systems for human-robot collaboration, Procedia CIRP 72 (2018) 111–116.
[5] R.-G. Sandra, V.M. Becerra, J.R. Llata, E. Gonzalez-Sarabia, C. Torre-Ferrero, J. Perez-Oria, Working together: a review on safe human-robot collaboration in industrial environments, IEEE Access 5 (2017) 26754–26773.
[6] ISO/TS 15066:2016 Robots and Robotic, International Organization for Standardization, 2016.
[7] M. Bdiwi, M. Pfeifer, A. Sterzing, A new strategy for ensuring human safety during

various levels of interaction with industrial robots, CIRP Ann. 66 (2017) 453–456.
[8] E. Mariotti, E. Magrini, A.D. Luca, Admittance control for human-robot interaction using an industrial robot equipped with a f/t sensor, International Conference on Robotics and Automation (ICRA), 2019.
[9] P. Tsarouchi, S. Makris, G. Chryssolouris, Human-robot interaction review and challenges on task planning and programming, Int. J. Comput. Integr. Manuf. (IJCIM) 29 (2016) 916–931.
[10] H. Liu, L. Wang, An AR-based worker support system for human-robot collaboration, Proc. Manuf. 11 (2017) 22–30.
[11] C. Vogel, C. Walter, N. Elkmann, Safeguarding and supporting future human-robot cooperative manufacturing processes by a projection-and camera-based technology, Proc. Manuf. 11 (2017) 39–46.
[12] C. Gkournelos, P. Karagiannis, N. Kousi, G. Michalos, S. Koukas, S. Makris, Application of wearable devices for supporting operators in human-robot cooperative assembly tasks, Proc. CIRP 76 (2018) 177–182.
[13] J. Gea Fernández, D. Mronga, M. Günther, T. Knobloch, M. Wirkus, M. Schröer, M. Trampler, S. Stiene, E. Kirchner, V. Bargsten, Multimodal sensor-based whole-body control for human-robot collaboration in industrial settings, Rob. Auton. Syst. 94 (2017) 102–119.
[14] E. Magrini, F. Ferraguti, A.J. Ronga, F. Pini, A. De Luca, F. Leali, Human-robot coexistence and interaction in open industrial cells, Rob. Comput.-Integr. Manuf. (RCIM) 61 (2020) 120–143.
[15] R.K. Ganesan, Y.K. Rathore, H.M. Ross, H.B. Amor, Better teaming through visual cues: how projecting imagery in a workspace can improve human-robot collaboration, IEEE Rob. Autom. Mag. 25 (2018) 59–71.
[16] Safety Requirements For Industrial robots, ISO 10218-1:2011, International Organization for Standardization, 2011.
[17] Robots For Industrial environments, ISO 10218-1:2006, International Organization for Standardization, 2006.
[18] J.A. Marvel, Performance metrics of speed and separation monitoring in shared workspaces, IEEE Trans. Autom. Sci. Eng. 10 (2013) 405–414.
[19] C. Sloth, H.G. Petersen, Computation of safe path velocity for collaborative robots, International Conference on Intelligent Robots and Systems (IROS), 2018.
[20] G. Michalos, N. Kousi, P. Karagiannis, C. Gkournelos, K. Dimoulas, S. Koukas, K. Mparis, A. Papavasileiou, S. Makris, Seamless human robot collaborative assembly: an automotive case study, Mechatronics 55 (2018) 194–211.
[21] G. Michalos, S. Makris, P. Tsarouchi, T. Guasch, D. Kontovrakis, G. Chryssolouris, Design considerations for safe human-robot collaborative workplaces, Proc. CIRP 37 (2015) 248–253.
[22] P.A. Lasota, T. Fong, J.A. Shah, A survey of methods for safe human-robot interaction, Found. Trends Rob. 5 (2017) 261–349.
[23] SafetyEYE, Pilz GmbH & Co., 2014.
[24] S. Kock, J. Bredahl, P.J. Eriksson, M. Myhr, K. Behnisch, Taming the robot better safety without higher fences, ABB Rev. (2006) 11–14.
[25] F. Vicentini, M. Giussani, L.M. Tosatti, Trajectory-dependent safe distances in human-robot interaction, Emerging Technology and Factory Automation (ETFA), 2014.
[26] C. Vogel, M. Poggendorf, C. Walter, N. Elkmann, Towards safe physical human-robot collaboration: a projection-based safety system, International Conference on Intelligent Robots and Systems (IROS), 2011.
[27] E. Kim, R. Kirschner, Y. Yamada, S. Okamoto, Estimating probability of human hand intrusion for speed and separation monitoring using interference theory, Rob. Comput.-Integr. Manuf. (RCIM) 61 (2020) 80–95.
[28] C. Byner, B. Matthias, H. Ding, Dynamic speed and separation monitoring for collaborative robot applications–concepts and performance, Rob. Comput.-Integr. Manuf. (RCIM) 58 (2019) 239–252.
[29] J.A. Marvel, R. Norcross, Implementing speed and separation monitoring in collaborative robot workcells, Rob. Comput.-Integr. Manuf. (RCIM) 44 (2017) 144–155.
[30] J.-.H. Chen, K.-.T. Song, Collision-Free motion planning for human-robot collaborative safety under cartesian constraint, International Conference on Robotics and Automation (ICRA), 2018.
[31] M.P. Polverini, A.M. Zanchettin, P. Rocco, A computationally efficient safety assessment for collaborative robotics applications, Robot. Comput. Integr. Manuf. 46 (2017) 25–37.
[32] F. Flacco, T. Kröger, A. De Luca, O. Khatib, A depth space approach to human-robot collision avoidance, International Conference on Robotics and Automation (ICRA), 2012.
[33] F. Fabrizio, A. De Luca, Real-time computation of distance to dynamic obstacles with multiple depth sensors, IEEE Rob. Autom. Lett. 2 (2016) 56–63.
[34] M. Cefalo, E. Magrini, G. Oriolo, Parallel collision check for sensor based real-time motion planning, International Conference on Robotics and Automation (ICRA), 2017.
[35] X. Zhao, J. Pan, Considering human behavior in motion planning for smooth human-robot collaboration in close proximity, International Symposium on Robot and Human Interactive Communication (RO-MAN), 2018.
[36] H. Rajnathsing, C. Li, A neural network based monitoring system for safety in shared work-space human-robot collaboration, Industr. Rob. 45 (2018) 481–491.
[37] H. Liu, T. Fang, T. Zhou, Y. Wang, L. Wang, Deep learning-based multimodal control interface for human-robot collaboration, Procedia CIRP 72 (2018) 3–8.
[38] P. Wang, H. Liu, L. Wang, R.X. Gao, Deep learning-based human motion recognition for predictive context-aware human-robot collaboration, CIRP Ann. 67 (2018) 17–20.
[39] S.A. Green, M. Billinghurst, X. Chen, J.G. Chase, Human-robot collaboration: a literature review and augmented reality approach in design, Int. J. Adv. Rob. Syst. (IJARS) 5 (2008) 1.
[40] S. Sato, S. Sakane, A human-robot interface using an interactive hand pointer that projects a mark in the real work space, International Conference on Robotics and Automation (ICRA), 2000.
[41] R.S. Andersen, O. Madsen, T.B. Moeslund, H.B. Amor, Projecting robot intentions into human environments, International Symposium on Robot and Human Interactive Communication (RO-MAN), 2016.
[42] R.C. Arkin, T.R. Collins, Skills Impact Study For Tactical Mobile Robot Operational Units, Georgia Institute of Technology, 2002.
[43] E. Rosen, W. David, P. Elizabeth, C. Gary, T. James, K. George, T. Stefanie, Communicating and controlling robot arm motion intent through mixed-reality head-mounted displays, Int. J. Rob. Res. 38 (2019) 1513–1526.
[44] D.Q. Huy, I. Vietcheslav, G.S.G. Lee, See-through and spatial augmented reality-a novel framework for human-robot interaction, International Conference on Control, Automation and Robotics (ICCAR), 2017.
[45] J. Elsdon, Y. Demiris, Augmented reality for feedback in a shared control spraying task, International Conference on Robotics and Automation (ICRA), 2018.
[46] B. M., F.C. Park, Robot sensor calibration: solving AX = XB on the Euclidean groupc, Trans. Rob. Autom. (1994).
[47] F. Y., R.L. Graham, Finding the convex hull of a simple polygon, J. Algo. 4 (4) (1983) 324–331.
[48] R.B. Rusu, Semantic 3d object maps for everyday manipulation in human living environments, Künstliche Intelligenz 24 (4) (2010) 345–348.
[49] C. Vogel, C. Walter, N. Elkmann, A projection-based sensor system for safe physical human-robot collaboration, International Conference on Intelligent Robots and Systems (IROS), 2013.
[50] M. Ivan, K. Joni-Kristian, L. Lasse, Projector calibration by inverse camera calibration, Scandinavian Conference on Image Analysis (SCIA), 2011.
[51] Safety of Machinery, ISO 12100:2010, International Organization for Standardization, 2015.
[52] G. Salvendy, Handbook of Human Factors and Ergonomics, John Wiley & Sons, 2012.
[53] E.L. Deci, R.J. Vallerand, L.G. Pelletier, R.M. Ryan, Motivation and education: the self-determination perspectivec, Educ. Psychol. 26 (3–4) (1991) 325–346.

PUBLICATION VI

Object Pose Estimation in Robotics Revisited

A. Hietanen, J. Latokartano, A. Foi, R. Pieters, V. Kyrki, M. Lanz and J.-K. Kämäräinen

arXiv preprint arXiv:1906.02783 (2020)

Object Pose Estimation in Robotics Revisited

Antti Hietanen[a,b,*], Jyrki Latokartano[b], Alessandro Foi[a], Roel Pieters[b], Ville Kyrki[c], Minna Lanz[b], Joni-Kristian Kämäräinen[a]

[a] *Computing Sciences, Tampere University, Finland*
[b] *Automation Technology and Mechanical Engineering, Tampere University, Finland*
[c] *Department of Electrical Engineering and Automation, Aalto University, Finland*

Abstract

Vision based object grasping and manipulation in robotics require accurate estimation of object's 6D pose. The 6D pose estimation has received significant attention in computer vision community and multiple datasets and evaluation metrics have been proposed. However, the existing metrics measure how well two geometrical surfaces are aligned - ground truth vs. estimated pose - which does not directly measure how well a robot can perform the task with the given estimate. In this work we propose a probabilistic metric that directly measures success in robotic tasks. The evaluation metric is based on non-parametric probability density that is estimated from samples of a real physical setup. During the pose evaluation stage the physical setup is not needed. The evaluation metric is validated in controlled experiments and a new pose estimation dataset of industrial parts is introduced. The experimental results with the parts confirm that the proposed evaluation metric better reflects the true performance in robotics than the existing metrics.

Keywords: Object pose estimation, robotics, grasping, 6D object pose, manipulation, probabilistic models, cognitive robotics

2010 MSC: 00-01, 99-00

*Corresponding author. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825196.

1. Introduction

One of the most common application in robotics is object manipulation where the fundamental task is to interact with objects in the environment. Succeeding in a such task requires accurate positioning of the robot end effector respect to the object, especially when interacting with objects having complex shape. In the literature, lot of works have focused on identifying and generating robust grasp pose hypothesis around a previously unseen object. Most of the recent methods rely on learning-based techniques, such as Convolutional Neural Networks (CNN) [1, 2, 3, 4], which allow learning of features from visual input that correspond to good quality grasps. More similarly to our work, probabilistic frameworks for grasp pose detection has been proposed in [5] where the grasp affordance model is generated by trial-and-error exploration and using the geometric properties of the 3D object. However, getting the object from the bin into the gripper in some manner does not guarantee successful precision manipulation or wrenching. Moreover, in industrial assembly, the objects are known before hand and the whole task is implemented based on a single object-related grasp pose which is selected by an experienced engineer. In this scenario, the estimated 6D pose of an object has the biggest contribution to grasp quality and eventually to whole task attempt.

Vision-based object recognition and 6D pose estimation from RGB-D input have recently become an active research topic in computer vision [6, 7]. In a typical workflow, a method first recognizes the object in a scene using RGB input and then estimates and refines the pose using depth (D) which provides a 3D point cloud; the method output is a 6D object pose with respect to the world coordinate frame. The methods are trained and optimized using training samples with ground truth pose annotations. Several 6D pose estimation datasets have been recently proposed [8, 9, 10, 11] for method comparison. The two most popular performance metrics are *average absolute translation/orientation error* and average distance of corresponding model points (ADC), calculated using the ground truth and estimated poses. A significant limitation of these metrics

is that they effectively measure only the difference between two transformation matrices but this difference is not necessarily indicative of the success in any particular task with a real robot. The robot vision community benefits from datasets and evaluation metrics that measure the actual success in real tasks without requiring physical setups.

The present work aims at providing a proper evaluation metric for robotic pose estimation and a demonstration dataset constructed using the proposed metric and required procedures. Specifically, we introduce a new benchmark dataset and performance metric for evaluating 6D pose estimation methods in robotics. The proposed benchmark does not require replication of the physical setup and yet it provides performance numbers valid for real tasks on real setups. The benchmark dataset consists of 3D models of industry relevant objects and approximately 600 test scenes with various amounts of clutter and occlusions. The provided performance metric is based on a conditional probability model that encodes the properties of the particular assembly task and measures the success in the task for estimated object poses.

The main contributions of this work are:

- A statistical formulation of a successfully conducted robotic task ($X=1$) given the estimated object pose. Concretely, the estimated object pose is converted and parametrized as 6D pose $\hat{\boldsymbol{\theta}}$ of the robot gripper in the object-relative coordinate space and evaluated using a conditional probability metric $P(X=1|\hat{\boldsymbol{\theta}})$. Interpretation of the metric is intuitive: 0.9 means that on average ninety out of one hundred attempts succeed with the given pose estimate. The 6D pose vector $\hat{\boldsymbol{\theta}}$ belongs to the 6D space $\mathcal{E}=\mathbb{R}^3\times S^3$, where S^3 denotes the 3D sphere parametrized by hyperspherical coordinates. Practical grasp probabilities are computed using non-parametric kernel regression on a number of collected random samples in $\mathcal{E}$ with the physical setup.
- An algorithm to generate automatically a large number of random samples for estimating the evaluation probabilities. The algorithm is validated

with several real setups where random samples are generated by a robot arm in industrial assembly tasks and with different grippers and objects. Sample success or failure ($X = \{0, 1\}$) is automatically detected to generate thousands of samples in 24 hours (video example [1]).

- A public benchmark for 6D object pose estimation in robotics. The benchmark consists of object models and test scenes with ground truth pose annotations and pre-computed probability models for each task configuration. In the experimental part of the work, the benchmark is used to evaluate several baseline and recent pose estimation methods.

It should be noted that the users of our benchmark do not need the physical setups to evaluate their methods and all performance numbers are still valid for the real setup in our laboratory. On the other hand, the proposed sampling procedures can be used to construct novel benchmarks with different physical setups in other laboratories. All code and data will be made publicly available to facilitate fair comparisons and to promote pose estimation research in robotics.

2. Related Work

Section 2.1 provides a brief review of the existing pose estimation datasets and their performance metrics, and Section 2.2 introduces popular baseline and more recent algorithms for 6D object pose estimation.

2.1. Benchmark datasets and performance metrics

Our main focus is on pose estimation from 3D data which is today easily available due to good quality and inexpensive RGB-D (color + depth) sensors. The authors acknowledge that there are numerous works dealing with 3D recognition and pose estimation from 2D input such as gray level or color images. There are also many available "2D-to-3D" benchmark datasets such as Pascal3D [12]. However, for practical robot manipulation RGB is often too limited setting and 3D sensing can be readily adopted.

[1] `https://youtu.be/g4e_-p4fTEI`

The early works did evaluations on 3D object models from 3D scan datasets and synthetic scenes. A popular dataset is the Stanford 3D Scanning Repository [13] that contains the famous Stanford Bunny. For these datasets the typical performance metrics are *3D translation and 3D rotation errors* [14].

One of the first real and still widely used 3D object recognition and 6D pose estimation datasets is LineMod introduced by Hinterstoisser et al. [11]. LineMod training data consists of 3D models and turn-table captured RGB and depth images. The test data consists of various cluttered scenes that were capture from multiple view points on a turn-table. As a unified performance metric Hinterstoisser et al. proposed to use the ADC metric which calculates the distance between model points transformed by the ground truth pose and the estimated pose. The metric is intuitive as it directly evaluates the fit of the two surfaces. However the metric is not well defined for objects having symmetric properties. Hinterstoisser proposed a metric where the distance between the corresponding points were replaced with the distance to the closest point and thus avoiding the symmetry problem.

Recently, Hodan et al. [8] introduced Benchmark for 6D Object Pose Estimation (BOP). BOP contains eight similarly captured datasets, including LineMod, that span various kinds of objects and scenes from household objects and scenes [15] to industrial [16]. Hodan et al. evaluated 15 recent methods on all eight datasets using a unified evaluation protocol. Their evaluation protocol takes into account view point dependent pose uncertainty and therefore they adopted the *Visible Surface Discrepancy* (VSD) [10] as the main error metric. VSD is invariant to pose ambiguity, i.e. due to the object symmetry there can be multiple poses that are indistinguishable. However the method requires additional ground truth in the form of visibility masks.

All above datasets and metrics measure the pose error as the misalignment between the ground truth and estimated object surface points. This requirement is important, for example, in augmented reality applications where the perceived virtual object must align well with the real environment. However, in robotics the performance metric should measure success in the target tasks

such as industrial assembly or disassembly.

2.2. 6D pose estimation methods

Sate-of-the-art methods divide RGB-D object pose estimation into two stages [7, 6]: i) detection of objects from RGB and ii) detected object pose estimation from depth (point cloud). Object detection is out of the scope of this work and therefore we briefly discuss the methods in the recent evaluation by Yang et al. [9] with their codes available (note that NNSR is a robustified version of SS).

Random Sample Consensus (RANSAC). RANSAC is a widely used technique for 6D pose estimation [17, 18, 19] adopted from the 2D domain. It is an iterative process that uses random sampling technique to generate candidate solutions for a model (transformation) that aligns two surfaces with a minimum point-wise error. Free parameter of the method is N_{RANSAC} which is the maximum count of pose hypothesis the algorithm samples matches from the correspondence set. The algorithm iteratively samples candidate transformations which are evaluated by transforming all the matched points and calculating the Euclidean distance between the corresponding points. All transformed point matches with distance less than d_{RANSAC} are counted as inliers. The final pose is estimated using all inlier points for transformation with the largest number of inliers.

Hough Transform (HG). Hough transform [20] is an alternative to RANSAC; instead of random samples each point match casts votes and pose with the largest number of votes is selected. There are several methods adopting this principle [21, 22] and for the experiments the Hough Grouping (HG) method by Tombari et al. [22] was selected. For fast computation, the method uses a unique model reference point (mass centroid) and bins represent pose around the reference point. To make correspondence points invariant to rotation and translation between the model and scene, every point is associated with a local reference frame [23]. The main parameter of the method is the pose bin size - coarse size provides faster computation but increases pose uncertainty.

Spectral Technique (ST). Leordeanu and Hebert [24] proposed a spectral grouping technique to find coherent clusters from the initial set of feature matches. The method takes into account the relationship between points and correspondences and finally uses an eigen-decomposition to estimate the confidence of a correspondence to be an inlier.

First the algorithm creates an affinity matrix $\boldsymbol{M}$ which entries represent weighted links between correspondences. The weights are estimated by calculating the pairwise similarity between two correspondences using a rigidity constraint:

$$M(c_i, c_j) = \min \left\{ \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\|\boldsymbol{x}'_i - \boldsymbol{x}'_j\|}, \frac{\|\boldsymbol{x}'_i - \boldsymbol{x}'_j\|}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|} \right\} , \tag{1}$$

where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are the model and captured scene 3D points, respectively. The diagonal elements of the matrix measure the level of individual assignments i.e. how well f_i and $f_i^{'}$ match. After computing $\boldsymbol{M}$, the principle eigenvector $\boldsymbol{v}$ of $\boldsymbol{M}$ is calculated and the location of the maximum value v_i gives the highest confidence of c_i being in the inlier set. Next, all the correspondences conflicting with c_i are removed from the initial set of matches $\boldsymbol{C}$ and procedure is repeated until $v_i = 0$ or $\boldsymbol{C}$ is empty and finally the generated inlier set is returned.

Geometric Consistency (GC). While the RANSAC and Hough transform based methods operate directly on the 3D points there are methods that exploit the local neighborhood of points to establish more reliable matches between model and scene point clouds [25, 26]. Geometric Consistency Grouping (GC) [25] is a strong baseline and it has been implemented in several point cloud libraries. GC works independently from the feature space and utilizes only the spatial relationship of the corresponding points. The algorithm evaluates the consistency of two correspondences c_i and c_j using a compatibility score

$$d(c_i, c_j) = \Big| \|\boldsymbol{x}_i - \boldsymbol{x}_j\| - \big\|\boldsymbol{x}'_i - \boldsymbol{x}'_j\big\| \Big| < \tau_{\mathrm{GC}} \ . \tag{2}$$

GC simply measures distances near the points and assigns correspondences to the same cluster if their geometric inconsistency is smaller than the threshold value τ_{GC}.

GC is initialized with a fixed number of clusters each having a seed correspondence. Then for each cluster it iteratively searches correspondences which satisfy the compatibility score (2), mark them as visited and continue the process until all the correspondences are visited. Finally, all the cluster sets can be optionally refined using RANSAC. In principle, the GC algorithm can return more than one cluster and for pose estimation the cluster with the largest number of correspondences is used as the pose estimate [27].

Search of Inliers (SI). A recent method by Buch et al. [26] achieves state-of-the-art on several benchmarks. It uses two consecutive processing stages, local voting and global voting. The first voting step performs local voting, where locally selected correspondence pairs are selected between a model and scene, and the score is computed using their pair-wise similarity score $s_L(c)$. At the global voting stage, the algorithm samples point correspondences, estimates a transformation and gives a global score to the points correctly aligned outside the estimation point set: $s_G(c)$. The final score $s(c)$ is computed by combining the local and global scores, and finally $s(c)$ are thresholded to inliers and outliers based on Otsu's bimodal distribution thresholding.

3. Evaluating Object Pose in Robotic

The standard procedure in industrial robotics is to manually set up and program the needed manipulation task. An experienced engineer is able to find a stable pose for grasping and select a gripper and fingers that are good for the given task. However, all settings are made with the assumption that object pose is accurate but which is difficult to achieve in practice even with the best computer vision methods.

A probabilistic formulation of success in the given task with a pose estimate is derived in Section 3.1. This formulation is used to define sampling procedures to construct a pose estimation benchmark for a physical setup (task) in Section 3.2. However, the users of a benchmark do not need the physical setup but only a set of test images, pose ground truth and the estimated probability function.

3.1. Probability of completing a programmed task $P(X=1)$

The success of a robot to complete its task is a binary random variable $X \in \{0,1\}$ where $X=1$ denotes a successful attempt and $X=0$ denotes an unsuccessful attempt (failure). Therefore, X follows the Bernoulli distribution, $P(X|p) = p^X(1-p)^{1-X}$, with complementary probability of success and failure: $E(X) = P(X=1) = 1 - P(X=0)$, where E denotes the mathematical expectation. The pose is defined by 6D pose coordinates $\boldsymbol{\theta} = (t_x, t_y, t_z, r_x, r_y, r_z)^T$ where the origin is the object centric coordinate frame. The translation vector $(t_x, t_y, t_z)^T \in \mathbb{R}^3$ and 3D rotation $(r_x, r_y, r_z)^T \in SO(3)$ both have three degrees of freedom. The rotation is in axis-angle representation, where the length of the 3D rotation vector is the amount of rotations in radians, and the vector itself gives the axis about which to rotate. Adding pose to the formulation makes the success probability a conditional distribution and expectation a conditional expectation. The conditional probability of a successful attempt is

$$p\left(\boldsymbol{\theta}\right) = E(X|\boldsymbol{\theta}) = P(X=1|\boldsymbol{\theta}) = 1 - P(X=0|\boldsymbol{\theta}) \ . \tag{3}$$

The maximum likelihood estimate of the Bernoulli parameter $p \in [0,1]$ from N homogeneous samples y_i, $i=1,\ldots,N$, is the sample average

$$\hat{p}_{\mathrm{ML}} = \frac{1}{N}\sum_{i=1}^{N} y_i \ , \tag{4}$$

where homogeneity means that all samples are realization of a common Bernoulli random variable with unique underlying parameter p. However, guaranteeing homogeneity would require that the samples $\{y_i, i=1,\ldots,N\}$ were either all collected at the same pose $\boldsymbol{\theta}_1 = \cdots = \boldsymbol{\theta}_N$, or for different poses that nonetheless yield same probability $p(\boldsymbol{\theta}_1) = \cdots = p(\boldsymbol{\theta}_N)$, i.e. it would require us either to collect multiple samples for each $\boldsymbol{\theta} \in SE(3)$ or to know beforehand p over $SE(3)$ (which is what we are trying to estimate). This means that in practice p must be estimated from non-homogeneous samples, i.e. from $\{y_i, i=1,\ldots,N\}$ sampled at pose $\{\boldsymbol{\theta}_i, i=1,\ldots,N\}$ which can be different and having different underlying $\{p(\boldsymbol{\theta}_i), i=1,\ldots,N\}$.

The actual form of p over $SE(3)$ is unknown and depends on many factors, e.g., the shape of an object, properties of a gripper and a task to be completed. Therefore it is not meaningful to assume any parametric shape such as the Gaussian or uniform distribution. Instead, we adopt the Nadaraya-Watson non-parametric estimator which gives the *probability of a successful attempt* as

$$\hat{p}_{\boldsymbol{h}}(\boldsymbol{\theta}) = \frac{\sum_{i=1}^{N} y_i K_{\boldsymbol{h}}(\boldsymbol{\theta}_i - \boldsymbol{\theta})}{\sum_{i=1}^{N} K_{\boldsymbol{h}}(\boldsymbol{\theta}_i - \boldsymbol{\theta})} \ , \tag{5}$$

where $\boldsymbol{\theta}_i$ denotes the poses at which y_i has been sampled and $K_{\boldsymbol{h}} : \mathcal{E} \to \mathbb{R}^+$ is a non-negative multivariate kernel with vector scale $\boldsymbol{h} = \left(h_{t_x}, h_{t_y}, h_{t_z}, h_{r_x}, h_{r_y}, h_{r_z}\right)^T > 0$.

In this work, $K_{\boldsymbol{h}}$ is the multivariate Gaussian kernel

$$\begin{aligned} K_{\boldsymbol{h}}(\boldsymbol{\theta}) = G\!\left(\tfrac{t_x}{h_{tx}}\right) G\!\left(\tfrac{t_y}{h_{ty}}\right) G\!\left(\tfrac{t_z}{h_{tz}}\right) \sum_{j\in\mathbb{Z}} G\!\left(\tfrac{r_x+2j\pi}{h_{rx}}\right) \cdot \\ \sum_{j\in\mathbb{Z}} G\!\left(\tfrac{r_y+2j\pi}{h_{ry}}\right) \sum_{j\in\mathbb{Z}} G\!\left(\tfrac{r_z+2j\pi}{h_{rz}}\right), \end{aligned} \tag{6}$$

where G is the standard Gaussian bell, $G(\theta) = (2\pi)^{-\frac{1}{2}} e^{\frac{1}{2}\theta^2}$. The three sum terms in (6) realize the modulo-2π periodicity of $SO(3)$.

The performance of the estimator (5) is heavily affected by the choice of $\boldsymbol{h}$, which determines the influence of samples y_i in computing $\hat{p}_{\boldsymbol{h}}(\boldsymbol{\theta})$ based on the difference between the estimated and sampled poses $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_i$. Indeed, the parameter $\boldsymbol{h}$ can be interpreted as reciprocal to the bandwidth of the estimator: too large $\boldsymbol{h}$ results in excessive smoothing whereas too small results in localized spikes.

To find an optimal $\boldsymbol{h}$, we use the leave-one-out (LOO) cross-validation method. Specifically, we construct the estimator on the basis of $N-1$ training examples leaving out the i-th sample:

$$\hat{p}^{\,\mathrm{LOO}}_{\boldsymbol{h}}(\boldsymbol{\theta}, i) = \frac{\sum_{j\neq i} y_j K_{\boldsymbol{h}}(\boldsymbol{\theta}_j - \boldsymbol{\theta})}{\sum_{j\neq i} K_{\boldsymbol{h}}(\boldsymbol{\theta}_j - \boldsymbol{\theta})} \ .$$

The likelihood of y_i given $\hat{p}^{\,\mathrm{LOO}}_{\boldsymbol{h}}(\boldsymbol{\theta}_i, i)$ is either $\hat{p}^{\,\mathrm{LOO}}_{\boldsymbol{h}}(\boldsymbol{\theta}_i, i)$ if $y_i = 1$, or $1 - \hat{p}^{\,\mathrm{LOO}}_{\boldsymbol{h}}(\boldsymbol{\theta}_i, i)$ if $y_i = 0$. We then select $\boldsymbol{h}$ that maximizes the total LOO log-likelihood over the whole set S_y:

$$\hat{\boldsymbol{h}} = \arg\max_{\boldsymbol{h}} \sum_{i|y_i=1} \log\left(\hat{p}^{\,\mathrm{LOO}}_{\boldsymbol{h}}(\boldsymbol{\theta}_i, i)\right) + \sum_{i|y_i=0} \log\left(1 - \hat{p}^{\,\mathrm{LOO}}_{\boldsymbol{h}}(\boldsymbol{\theta}_i, i)\right).$$

Our choices of the kernel and LOO optimization of the kernel parameters result to probability estimates that are verifiable by controlled experiments (as illustrated in Fig. 4).

3.2. Sampling the pose space

Section 3.1 provides us a formulation of the probability of successful robotic manipulation given the object relative grasp pose $P(X = 1|\boldsymbol{\theta})$. The practical realization of the probability values is based on Nadaraya-Watson non-parametric kernel estimator that requires a number of samples in various poses $\boldsymbol{\theta}_i$ and information of success $y_i = 1$ or failure $y_i = 0$ for each attempt. In this stage, a physical setup is needed for sampling, but the users of the benchmark do not need to replicate the setup - they need only the pre-computed probability densities provided with the benchmark. For practical reasons we make the following assumptions:

- We define a canonical grasp pose respect to a manipulated object which is select based on the object intrinsic parameters (i.e. the distribution of mass) and task requirements (i.e. on which way the object is being installed). During the sampling procedure the canonical pose is located using a 2D marker.
- We sample the pose space around the canonical grasp pose, and therefore $\boldsymbol{\theta} = (t_x, t_y, t_z, r_x, r_y, r_z)^T$ defines $SE(3)$ "displacement" from the canonical grasp pose. Sampling was started by first finding the sampling limits of each dimension and then sampling within the limits. The limits were found by manually guiding the end effector away from the canonical grasp pose along each dimension until the task execution always failed. The limits are listed in Table 1.

With the help of these assumptions we are able to define a sampling procedure that can record samples and their success or failures automatically. The main limitation of this approach is that the pose space is sampled only near the canonical grasp pose which is not guaranteed to be the best option in every

scenario. For instance, the grasp pose might be unreachable due to robots kinematic constraints or obstructing objects. In our work, we assume that the canonical grasp pose is always reachable.

Coordinate transformations. In the work, a coordinate transformation $\boldsymbol{T}_B^A$ denotes a 4×4 homogeneous transformation matrix that describes the position of the frame B origin and the orientation of its axes, relative to the reference frame A.

For a practical implementation used in our experiments the transformation components are (Figure 1):

- $\boldsymbol{T}_{grasp}^{marker}$ – a constant transformation from the canonical grasp pose to the marker frame;
- $\boldsymbol{T}_{marker}^{sensor}$ – computed transformation from the marker frame to the sensor frame;
- $\boldsymbol{T}_{sensor}^{effector}$ – a constant transformation from the sensor frame to the robot end effector frame (camera is attached to the end effector);
- $\boldsymbol{T}_{effector}^{world}$ – computed transformation from the end effector frame to the world frame (robot origin).

The world frame is fixed to the robot frame (i.e. center of the robot base) and programming is based on the tool point that is the end effector frame. The coordinate transformation $\boldsymbol{T}_{effector}^{world}$ can be automatically calculated using the joint angles and known kinematic equations. $\boldsymbol{T}_{sensor}^{effector}$ is computed using the standard procedure for hand-eye calibration with a printed chessboard pattern [28]. Automatic and accurate estimation of the object pose during the sampling is realized by attaching an artificial 2D markers to the manipulated objects (see Fig. 2 for an example). For a calibrated camera the ArUco library [29] provides an accurate real-time pose of the marker with respect to the sensor frame $\boldsymbol{T}_{marker}^{sensor}$. The constant offset $\boldsymbol{T}_{grasp}^{marker}$ from the marker to the actual grasp pose is object-marker specific and it is estimated manually by hand-guiding the end effector to the desired grasp location on the object (canonical grasp pose) and

then measuring the difference between this pose and the marker pose:

$$\boldsymbol{T}_{grasp}^{marker} = \left(\boldsymbol{T}_{marker}^{world}\right)^{-1} \boldsymbol{T}_{grasp}^{world}.$$

During the sampling procedure, the canonical grasp pose is then calculated respect to world frame as:

$$\boldsymbol{T}_{grasp}^{world} = \boldsymbol{T}_{effector}^{world} \cdot \boldsymbol{T}_{sensor}^{effector} \cdot \boldsymbol{T}_{maker}^{sensor} \cdot \boldsymbol{T}_{grasp}^{marker} \tag{7}$$

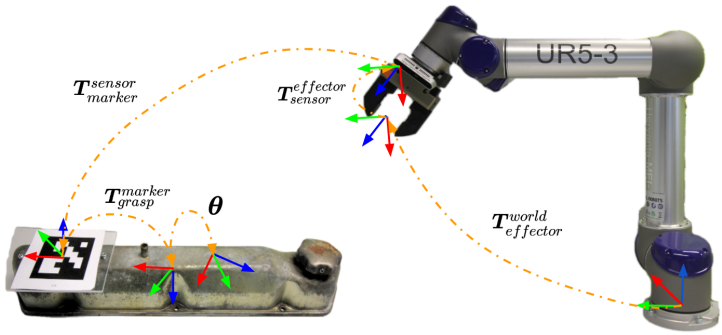


Figure 1: Coordinate frames used in random sampling of poses for assembly tasks.

Finally, samples around the canonical grasp pose are generated from

$$\hat{\boldsymbol{T}}_{grasp}^{world} = \boldsymbol{T}_{grasp}^{world} \cdot \boldsymbol{\Phi}(\boldsymbol{\theta}) \tag{8}$$

where the operator $\boldsymbol{\Phi}(\cdot)$ converts the 6D pose vector to a 4×4 matrix representation

$$\boldsymbol{\Phi}(\boldsymbol{\theta}) = \begin{bmatrix} \boldsymbol{R}_{3\times3} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{9}$$

The generated pose sample is defined in the vicinity of the canonical pose by the translation shift $\boldsymbol{t} = (t_x, t_y, t_z)^T$ and rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ constructed from the axis-angle vector $(r_x, r_y, r_z)^T$.

Detection of failures. Each of the manipulated object has a predefined position and orientation how it should be installed, i.e. *ground truth installation pose*, respect to the target object. For instance most of the motor parts have to be placed on the motor block precisely in order to fasten the parts with screws. The robot task is to bring the part to this pose and finally release the part by opening the gripper fingers. In addition, using excessive force during the task can cause damage to the manipulated objects. In the work there are two sources of information for detecting manipulation failures:

- too large difference between the installation pose of the manipulated object and the corresponding ground truth and
- too large wrench torque at the end effector at any moment of task execution (e.g. due to collisions), including grasping, carrying and installation.

Thresholds for the above are task specific and in our experiments they were manually set based on preliminary experiments.

For evaluation the success of the part installation in the terms of correct location the two thresholds are used: τ_t for the maximum translation error and τ_r for the maximum orientation error (both task specific). These are computed using installation pose $\hat{\mathbf{\Gamma}} = \left[\hat{\boldsymbol{R}} \mid \hat{\boldsymbol{t}}\right]$ measured using the marker attached to the manipulated object and the ground truth installation pose $\mathbf{\Gamma} = [\boldsymbol{R} \mid \boldsymbol{t}]$. Both are measured respect to the target object on which the manipulated object is being installed. The installation was successful if

$$\begin{aligned} &\left\|\boldsymbol{t} - \hat{\boldsymbol{t}}\right\| \leq \tau_t \\ &\arccos\left(\frac{\operatorname{trace}\left(\hat{\boldsymbol{R}}\boldsymbol{R}^{-1}\right) - 1}{2}\right) \leq \tau_r \end{aligned} \quad . \tag{10}$$

The torque is used to detect if the robot collides with its environment during the task execution. In addition, if the robot places the object to the correct position with too high wrench the whole task is considered as an unsuccessful attempt. The external wrench is computed based on the error between the joint torques required to stay on the programmed trajectory and the expected joint torques.

The robot's internal sensors provide the torque measurements $\boldsymbol{F} = (f_x, f_y, f_z)$, where f_x, f_y and f_z are the forces in the axes of the robot frame coordinates and measured in Newtons. For each task the limit f_{max} was manually set for each operation stage using preliminary experiments and violating the threshold, i.e $\|\boldsymbol{F}\| > f_{max}$, was recorded as failure. All sampling steps are in Algorithm 1.

Algorithm 1: Practical sampling of the pose space

Input: Robot program waypoints $\mathcal{W} := \{w_i | i = 1, \ldots, N\}$; Number of samples S

Output: Set of samples $\{(\boldsymbol{\theta}_i, y_i) | i = 1, \ldots, S\}$

for $i = 1$ **to** S **do**
- $y_i \leftarrow$ success;
- $\boldsymbol{\theta}_i \leftarrow$ SampleRandomDisplacement();
- $\boldsymbol{T}^{sensor}_{marker} \leftarrow$ DetectMarker($\mathcal{W}$);
- $\boldsymbol{T}^{world}_{sensor} \leftarrow$ ComputeForwardKinematics();
- `// end effector pose in object (marker) coordinate system`
- $\hat{\boldsymbol{T}}^{marker}_{grasp} \leftarrow$ SamplePose($\boldsymbol{\theta}_i, \boldsymbol{T}^{marker}_{grasp}$);
- `// end effector pose in world coordinate system`
- $\hat{\boldsymbol{T}}^{world}_{grasp} \leftarrow \boldsymbol{T}^{world}_{sensor} \cdot \boldsymbol{T}^{sensor}_{marker} \cdot \hat{\boldsymbol{T}}^{marker}_{grasp}$;
- GraspObject($\hat{\boldsymbol{T}}^{world}_{grasp}, \mathcal{W}$);
- **if** *SuccessfulGrasp() is False* **then**
  - `// marker detected on the table or force limits exceeded`
  - $y_i \leftarrow$ failure
- **else**
  - InstallObject($\mathcal{W}$);
  - **if** *SuccessfulInstall() is False* **then**
    - `// marker on wrong pose or force limits exceeded`
    - $y_i \leftarrow$ failure
- Record($\boldsymbol{\theta}_i$, y_i);
- MoveObjectToStart($\mathcal{W}$);

4. Experiments

We implemented four assembly tasks for a robot arm. For each task the gripper and custom made fingers were used. To accurately estimate the success probabilities, a large number of pose samples were needed for each task. For that reason, the setups were made autonomous so that the task success was automatically detected. This was achieved by verifying the final pose of the assembled parts and measuring the torque sensor readings during the task execution (Section 3.2). The tasks, robot setups, experimental results and verification experiments are explained in the following.

4.1. Tasks

To conduct experiments on practical tasks they were selected from the production line of a local engine manufacturing company. The selected tasks were: (Task 1) installation of a motor cap 1, (Task 2) installation of a motor frame and (Task 3) installation of a motor cap 2 (different engine model). The fourth task (Task 4) is different from others: picking and dropping a part to a container (the *faceplate* part from the Cranfield assembly benchmark). As Task 4 does not require precise manipulation, the task requires less accurate pose than the others. This can be verified in Table 1 where the Task 4 limits are less strict (by order of magnitude) as compared to the other tasks. Cranfield faceplate was selected since its 3D model is publicly available and the part is used in robot manipulation studies. The tasks were programmed by an experienced engineer who also carefully selected the grippers and fingers. The engineer was instructed that accurate pose is always available.

4.2. Setup

In Fig. 2 is illustrated the robotic setup used in our experiments. The setup consisted of a model 5 Universal Robot Arm (UR5) and a Schunk PGN-100 gripper. The gripper operates pneumatically and was configured to have a high gripping force (approximately 600N) to prevent object slippage. In addition, the gripper had custom 3D printed fingers plated with rubber. For visual perception,

an Intel RealSense D415 RGB-D sensor was secured on a 3D printed flange mounted between the gripper and the robot end effector. All the in-house made 3D prints were made using nylon reinforced with carbon fiber to tolerate external forces during the experiments. The computation was performed on a single laptop with Ubuntu 18.04. All tasks and the canonical grasp poses were validated by executing the task 100 times with pose obtained using the 2D patterns (Section 3.2). No failures occurred during the validation. On average, successful executions took 45-55 seconds and in 24 hours the robot was able to execute approximately 1,100 attempts. The setup was able automatically to recover from most of the failure cases (dropping the object, object collision, etc.), however, if the marker was occluded by the environment or if the manipulated object got jammed against internal parts of the motor, the system was restarted by a human operator.

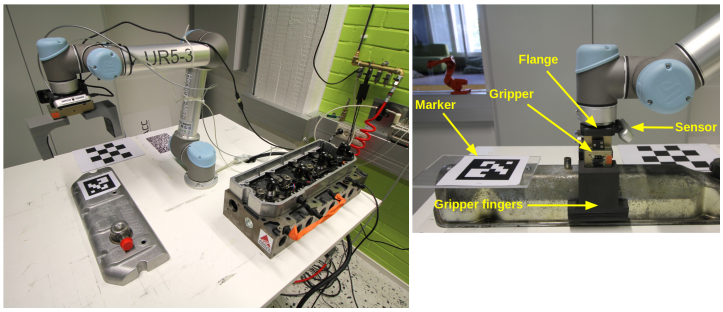


Figure 2: The experimental robot setup to sample the pose space of the engine cap 1. The task is to grasp and accurate assembly the cap to the engine block. Failures in task execution were automatically detected during sampling (Section 3.2).

4.3. RGB-D dataset

In the dataset each of the object models are stored as a point cloud that represents a set of N 3D points $\{\boldsymbol{x}_i | i = 1, \ldots, N\}$. In addition, for each point the corresponding color value $\boldsymbol{c} \in \mathbb{N}^3$ is stored. The models were generated

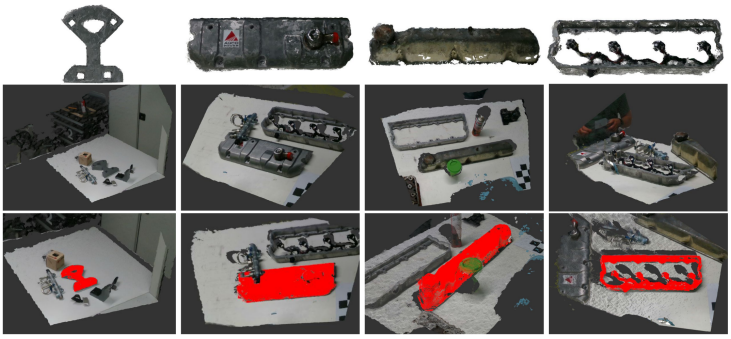


Figure 3: Top: Point cloud models of the used industry objects; faceplate, motor cap 1, motor cap 2 and a motor frame. The models were reconstructed by combining different view points of the robot arm and RGB-D sensor. Middle: example test samples (colored point clouds). Bottow: renderings of the object models on the test images using the ground truth poses.

using the same setup. The point cloud models were obtained automatically by moving the robot arm with the attached RGB-D sensor around each object. By using the camera poses obtained from robot kinematics the measurements were merged to a single point cloud that is the stored object model (see Fig. 3). The automatically captured point clouds were then manually checked and all artifacts and redundant parts of the reconstructed point cloud were removed manually using the open-source mesh processing software MeshLab [30]. Finally, the coordinate system of each model point cloud was aligned with the canonical grasp pose of the object.

The test dataset was generated in a similar manner by moving the arm around the objects. For each of the objects 150 test images were collected in three different settings: 1) a single target object present, 2) multiple objects present and 3) the target object partially occluded by other object(s). The dataset contains manually verified ground truth to align the model point cloud to each test image and further to locate the canonical grasp pose relative to sensor ($\boldsymbol{T}_{sensor}^{grasp}$).

Table 1: Sampling limits for translation (t_x, t_y, t_z) and rotation (r_x, r_y, r_z) in meters and degrees, respectively. Beyond these limits the task always fails.

Variable	*Task Name*			
	Task 1	Task 2	Task 3	Task 4
t_x	[-9.0, 9.0] $\cdot 10^{-3}$	[-6.0, 6.0] $\cdot 10^{-3}$	[-9.0, 9.0] $\cdot 10^{-3}$	[-6.5, 8.5] $\cdot 10^{-3}$
t_y	[-1.0, 1.0] $\cdot 10^{-3}$	[-3.0, 2.5] $\cdot 10^{-3}$	[-5.0, 6.0] $\cdot 10^{-3}$	[-2.1, 2.1] $\cdot 10^{-2}$
t_z	[-1.0, 5.0] $\cdot 10^{-3}$	[-2.0, 4.0] $\cdot 10^{-3}$	[-2.0, 5.0] $\cdot 10^{-3}$	[-1.2, 1.7] $\cdot 10^{-2}$
r_x	[-6.3, 6.3] $\cdot 10^{0}$	[-6.3, 6.3] $\cdot 10^{0}$	[-2.0, 1.0] $\cdot 10^{0}$	[-1.5, 1.5] $\cdot 10^{1}$
r_y	[-5.0, 5.0] $\cdot 10^{-1}$	[-2.5, 1.0] $\cdot 10^{0}$	[-2.0, 2.0] $\cdot 10^{0}$	[-1.5, 1.5] $\cdot 10^{1}$
r_z	[-5.0, 5.0] $\cdot 10^{-1}$	[-1.5, 1.5] $\cdot 10^{0}$	[-4.0, 4.0] $\cdot 10^{0}$	[-1.5, 1.5] $\cdot 10^{1}$

4.4. Model validation

The probability model $P(X = 1|\boldsymbol{\theta})$ in Section 3.1 was fitted using the sampling procedure in Section 3.2. For all tasks approximately 3,300 valid samples were generated around task canonical poses.

The estimated probability models were validated by sampling each dimension separately on grid points and executing the task ten times on each point with real robot. The averaged task success rate on real robot was then compared against the proposed models and the estimated probabilities matched well as can be seen in Fig. 4.

4.5. Methods

Comparison included the methods in Section 2.2. All methods input point clouds of the model and scene. The model and scene point clouds were downsampled to fixed resolutions using a regular voxel grid to limit the amount of data for processing. Depending on the density of the cloud the size of voxels was $0.5 - 1.0mm$. Since methods also exploit surface normals they were estimated using the standard least squares plane fitting on points in a small neighborhood. To further reduce the computational complexity in the matching stage, we avoided using all the surface points as local keypoints and only select a

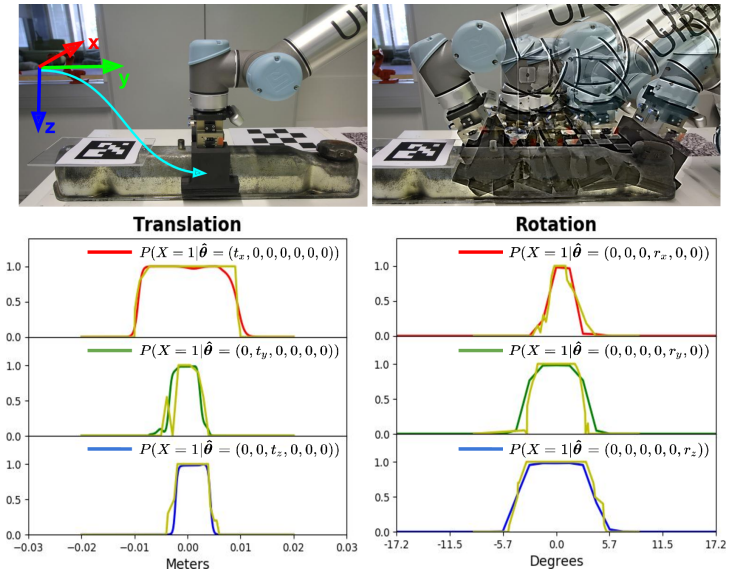


Figure 4: Motor cap 2 used in our Task 3. The coordinate system is object centric (top left) and pose samples are taken around a canonical grasp pose (see experiments for more details). Below are the estimated (the red, green and blue lines) and validated success probabilities (yellow line) on the six main axes (three translations and three rotations) in vicinity of the canonical grasp pose.

uniform subset of $1000 - 3000$ points per object model using the voxel grid filtering. Finally, local descriptors for point matching were computed using the local point neighborhoods. The SHOT [23] feature descriptor was selected since it performed the best in the preliminary experiments. The descriptor support radius was set to $0.125\times$ the object model's minimal bounding box diagonal. For each test scene, the best matching descriptors in L_2 sense between the model and scene were selected using a randomized *kd*-tree similarity search. The best matches formed then the initial set of correspondences for each method.

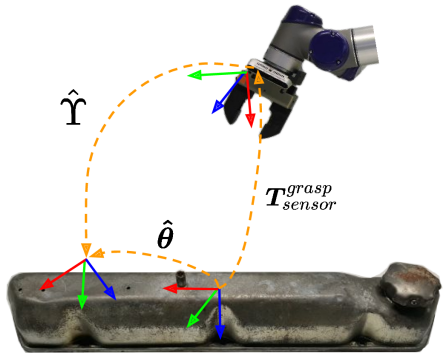


Figure 5: Coordinate frames in the evaluation procedure.

4.6. Performance indicators

The main performance metric in our work is the estimated success probability defined in Section 3.1. The probabilities were computed around the canonical grasp pose of each object and therefore the sampled values actually represent residual from this pose (see Fig. 5). The corresponding object-relative grasp pose of the pose estimate $\hat{\boldsymbol{\Upsilon}}$ is calculated as:

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\Phi}^{-1}\left(\boldsymbol{T}^{grasp}_{sensor}\,\hat{\boldsymbol{\Upsilon}}\right) \ , \tag{11}$$

where the transformation matrices $\boldsymbol{T}^{grasp}_{sensor}$ defines the canonical grasp pose respect to the sensor coordinate system. The $\boldsymbol{\Phi}^{-1}(\cdot)$ operator converts the 4×4 pose matrix to 6D vector representation. Finally, the task success is evaluated using the proposed metric as $P(X = 1|\hat{\boldsymbol{\theta}})$. We calculated the average probabilities over the whole dataset and also the proportion of images for which the probability is greater or equal to 0.90.

In addition to the proposed indicator we also report the ADC error calculated over the points transformed by the ground truth and estimated object pose as suggested in [11]. The ADC error is computed from

$$\boldsymbol{\epsilon}_{ADC} = \frac{1}{|\mathcal{M}|} \sum_{\boldsymbol{x} \in \mathcal{M}} \left\| \hat{\boldsymbol{\Upsilon}}\boldsymbol{x} - \boldsymbol{\Upsilon}\boldsymbol{x} \right\| \tag{12}$$

Table 2: Comparison of pose estimation methods with our dataset (single: single object in the scene; multi: multiple objects (clutter); occ: multiple objects and occlusion: all: average over all test samples).

Method	Task: *Task 1* Part: *Motor cap 1*; Gripper: *Shunker* Fingers: *Custom made* 							Task: *Task 2* Part: *Motor frame*; Gripper: *Shunker* Fingers: *Custom made* 						
	Average success probability				$\%\{p \geq 0.9\}$	Avg. ADC		Average success probability				$\%\{p \geq 0.9\}$	Avg. ADC	
	single	multi	occ	all	all	all	best-25%	single	multi	occ	all	all	all	best-25%
GC [25]	0.24	0.18	0.12	0.19	12%	0.08	$3.83{\cdot}10^{-3}$	0.21	0.22	0.19	0.21	9%	0.02	$5.36{\cdot}10^{-3}$
HG [22]	0.31	0.29	0.20	0.26	14%	0.06	$3.87{\cdot}10^{-3}$	0.28	0.27	0.27	0.28	15%	0.03	$5.19{\cdot}10^{-3}$
SI [26]	0.00	0.00	0.00	0.00	0%	0.46	$1.78{\cdot}10^{-1}$	0.14	0.04	0.03	0.07	5%	0.42	$1.81{\cdot}10^{-2}$
ST [24]	0.01	0.03	0.00	0.01	0%	0.35	$9.12{\cdot}10^{-2}$	0.23	0.16	0.07	0.17	7%	0.34	$4.38{\cdot}10^{-3}$
NNSR [20]	0.00	0.00	0.00	0.00	0%	0.26	$1.18{\cdot}10^{-1}$	0.00	0.00	0.00	0.00	0%	0.36	$1.50{\cdot}10^{-1}$
RANSAC [17]	0.00	0.00	0.00	0.00	0%	0.75	$1.71{\cdot}10^{-1}$	0.00	0.00	0.00	0.00	0%	0.65	$2.03{\cdot}10^{-1}$

Method	Task: *Task 3* Part: *Motor cap 2*; Gripper: *Shunker* Fingers: *Custom made* 							Task: *Task 4* Part: *Cranfield faceplate*; Gripper: *Shunker* Fingers: *Custom made* 						
	Average success probability				$\%\{p \geq 0.9\}$	Avg. ADC		Average success probability				$\%\{p \geq 0.9\}$	Avg. ADC	
	single	multi	occ	all	all	all	best-25%	single	multi	occ	all	all	all	best-25%
GC [25]	0.24	0.25	0.20	0.24	13%	0.09	$6.28{\cdot}10^{-3}$	0.66	0.67	0.59	0.64	65%	0.15	$4.57{\cdot}10^{-3}$
HG [22]	0.13	0.21	0.10	0.15	9%	0.11	$7.81{\cdot}10^{-3}$	0.64	0.68	0.56	0.63	60%	0.16	$3.43{\cdot}10^{-3}$
SI [26]	0.11	0.19	0.11	0.13	8%	0.09	$1.11{\cdot}10^{-2}$	0.37	0.43	0.20	0.35	35%	0.39	$9.94{\cdot}10^{-3}$
ST [24]	0.17	0.18	0.08	0.15	8%	0.11	$5.46{\cdot}10^{-3}$	0.40	0.39	0.30	0.37	36%	0.30	$6.47{\cdot}10^{-3}$
NNSR [20]	0.02	0.00	0.00	0.01	1%	0.19	$6.10{\cdot}10^{-2}$	0.05	0.04	0.07	0.05	5%	0.28	$7.16{\cdot}10^{-2}$
RANSAC [17]	0.00	0.00	0.00	0.00	0%	0.28	$1.24{\cdot}10^{-1}$	0.00	0.04	0.00	0.01	1%	0.51	$1.05{\cdot}10^{-1}$

where $\mathcal{M}$ is the set of model 3D points. We also report the top−25% ADC error, which is less affected by outliers.

4.7. Results

The results for all methods and parts are in Table 2. The two best methods are Hough Transform (HG) by Tombari et al. [22] and GC by Chen and Bhanu [25]. HG and GC perform considerably better than the two more state-of-the-art methods SI and ST although the performance of all methods remains surprisingly low. The two baselines, simple Hough voting (NNSR) and RANSAC, perform poorly.

Success probability vs. ADC. It is important to notice that the ADC results indicate clearly smaller difference between the methods than indicated by the success probability. The success probability measures directly performance in the physical task. This is even more evident in Fig. 6 of the ADC error and success probability graphs. The success probability is able to measure the points after which the success quickly drops from 1.0 to 0.0, but ADC (green points) regrades linearly even after these points and is thus uninformative. The non-linear behavior was verified in the controlled experiments in all the tasks as illustrated in Fig. 7. Moreover, the difference between the two metrics is further illustrated in the success probability vs. ADC scatter plots of all four tasks in Fig. 8.

5. Conclusions

This work addressed evaluation of vision based object pose estimation methods for robotics. In our experiments we demonstrated how the popular error measure, ADC, poorly indicates success in robot manipulation tasks and is therefore uninformative. As a novel solution, we proposed a probabilistic metric that measures the true success rate without the physical setup. The experimental results demonstrated poor performance of the existing methods which indicates that more work is needed for 6D object pose estimation in robotics. All data and code will be made publicly available to facilitate fair comparisons and to boost research on robot vision for vision based object grasping and manipulation.

References

[1] T.-T. Do, A. Nguyen, I. Reid, Affordancenet: An end-to-end deep learning approach for object affordance detection, in: 2018 IEEE international conference on robotics and automation (ICRA), IEEE, 2018, pp. 1–5.

[2] M. Gualtieri, A. Ten Pas, K. Saenko, R. Platt, High precision grasp pose detection in dense clutter, in: IROS, IEEE, 2016, pp. 598–605.

[3] L. Pinto, A. Gupta, Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours, in: ICRA, IEEE, 2016, pp. 3406–3413.

[4] J. Redmon, A. Angelova, Real-time grasp detection using convolutional neural networks, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2015, pp. 1316–1322.

[5] R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, J. Piater, Learning grasp affordance densities, Paladyn, Journal of Behavioral Robotics 2 (1) (2011) 1–17.

[6] F. Manhardt, W. Kehl, N. Navab, F. Tombari, Deep model-based 6d pose refinement in rgb, in: ECCV, 2018.

[7] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, N. Navab, Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again, in: ICCV, 2017.

[8] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, et al., Bop: benchmark for 6d object pose estimation, in: ECCV, 2018, pp. 19–34.

[9] J. Yang, K. Xian, Y. Xiao, Z. Cao, Performance evaluation of 3d correspondence grouping algorithms, in: 3DV, IEEE, 2017, pp. 467–476.

[10] T. Hodaň, J. Matas, Š. Obdržálek, On evaluation of 6d object pose estimation, in: ECCV, Springer, 2016, pp. 606–619.

[11] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, N. Navab, Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes, in: ACCV, Springer, 2012, pp. 548–562.

[12] Y. Xiang, R. Mottaghi, S. Savarese, Beyond pascal: A benchmark for 3d object detection in the wild, in: IEEE Winter Conference on Applications of Computer Vision (WACV), 2014.

[13] G. Turk, M. Levoy, Zippered polygon meshes from range images, in: SIGGRAPH, 1994.

[14] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, A. Fitzgibbon, Scene coordinate regression forests for camera relocalization in rgb-d images, in: CVPR, 2013, pp. 2930–2937.

[15] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, T.-K. Kim, Recovering 6d object pose and predicting next-best-view in the crowd, in: CVPR, 2016, pp. 3583–3592.

[16] T. Hodan, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, X. Zabulis, T-less: An rgb-d dataset for 6d pose estimation of texture-less objects, in: WACV, IEEE, 2017, pp. 880–888.

[17] M. A. Fischler, R. C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (6) (1981) 381–395.

[18] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, C. Rother, Learning 6d object pose estimation using 3d object coordinates, in: ECCV, Springer, 2014, pp. 536–551.

[19] E. Brachmann, F. Michel, A. Krull, M. Yang, S. Gumhold, C. Rother, Uncertainty-driven 6d pose estimation of objects and scenes from a single RGB, in: CVPR, 2016.

[20] P. V. Hough, Method and means for recognizing complex patterns, uS Patent 3,069,654 (Dec. 18 1962).

[21] J. Knopp, M. Prasad, G. Willems, R. Timofte, L. van Gool, Hough transform and 3D SURF for robust three dimensional classification, in: ECCV, 2010.

[22] F. Tombari, L. Di Stefano, Object recognition in 3d scenes with occlusions and clutter by hough voting, in: PSIVT, IEEE, 2010, pp. 349–355.

[23] F. Tombari, S. Salti, L. Di Stefano, Unique signatures of histograms for local surface description, in: ECCV, Springer, 2010, pp. 356–369.

[24] M. Leordeanu, M. Hebert, A spectral technique for correspondence problems using pairwise constraints, in: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Vol. 2, IEEE, 2005, pp. 1482–1489.

[25] H. Chen, B. Bhanu, 3d free-form object recognition in range images using local surface patches, Pattern Recognition Letters 28 (10) (2007) 1252–1262.

[26] A. Glent Buch, Y. Yang, N. Kruger, H. Gordon Petersen, In search of inliers: 3d correspondence by local and global voting, in: CVPR, 2014, pp. 2067–2074.

[27] A. Hietanen, J. Halme, A. G. Buch, J. Latokartano, J.-K. Kämäräinen, Robustifying correspondence based 6D object pose estimation, in: IEEE Int. Conf. on Robotics and Automation (ICRA), Singapore, 2017.

[28] F. C. Park, B. J. Martin, Robot sensor calibration: solving ax= xb on the euclidean group, IEEE Transactions on Robotics and Automation 10 (5) (1994) 717–721.

[29] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, M. J. Marín-Jiménez, Automatic generation and detection of highly reliable fiducial markers under occlusion, Pattern Recognition 47 (6) (2014) 2280–2292.

[30] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia, Meshlab: an open-source mesh processing tool., in: Eurographics Italian chapter conference, Vol. 2008, 2008, pp. 129–136.

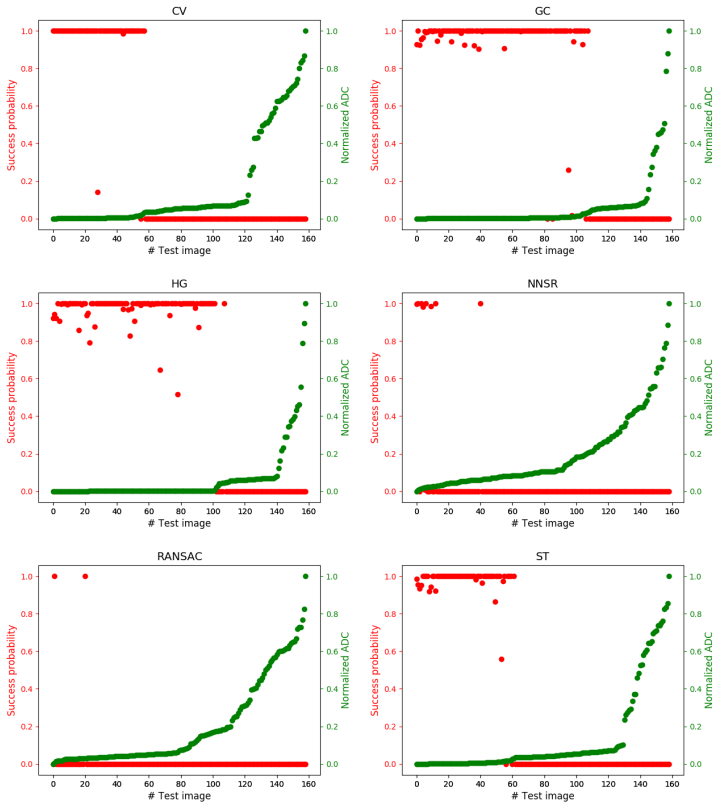


Figure 6: ADC pose error (green) and success probability (red) of all test images of Task 4 for different pose estimation methods. Images are sorted based on their ADC error. Note rapid change from success (1.0) to failure (0.0) when the error goes beyond certain points.

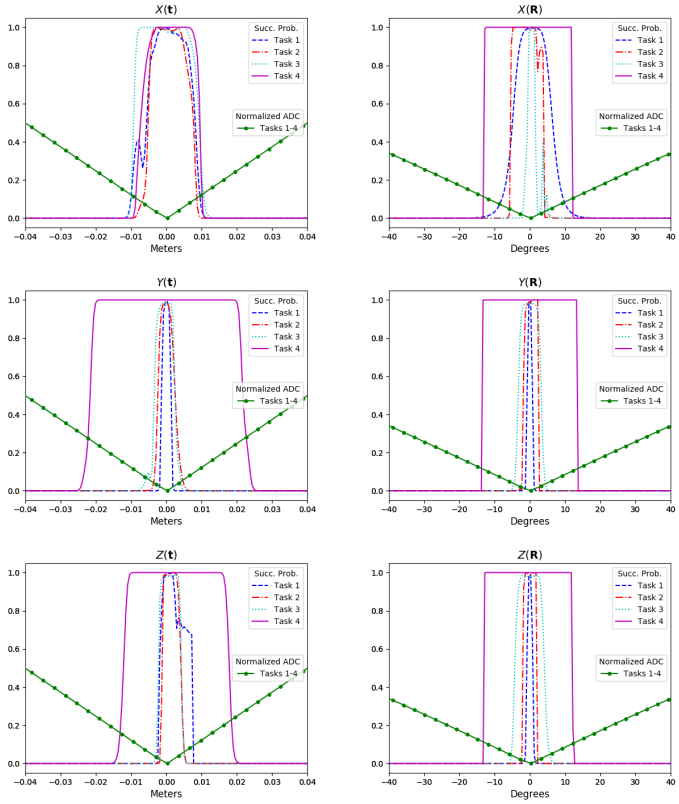


Figure 7: ADC and success probability from controlled experiments for all the tasks (Task 1 – Task 4). Effect of rotation (left column) and translation (right column) to the ADC and success probability.

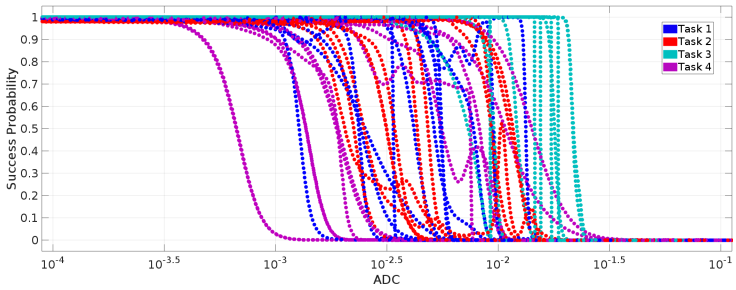


Figure 8: Success Probability vs. ADC scatterplot from controlled experiments for all the tasks (Task 1 – Task 4). The scatterplot shows that the ADC does not reflect the success probability, except for extreme and trivial cases of failure or success; the two measures cannot be put in correspondence to each other not even through a nonlinear mapping.