3D pose estimation for bin-picking: A data-driven approach using multi-light images

Submitted in partial fulfillment of the requirements for
the degree of

Doctor of Philosophy
in
Electrical and Computer Engineering

José Jerónimo Moreira Rodrigues

B.S., Electrical and Computer Eng., Instituto Superior Técnico, University of Lisbon
M.S., Electrical and Computer Eng., Instituto Superior Técnico, University of Lisbon

Carnegie Mellon University
Pittsburgh, PA

August, 2018

Doctoral Dissertation Committee

Professor Alexandre Bernardino

Instituto Superior Técnico, University of Lisbon

Professor Jianbo Shi

University of Pennsylvania

Professor João M. F. Xavier (Advisor)

Instituto Superior Técnico, University of Lisbon

Professor Marios Savvides

Carnegie Mellon University

Professor Pedro M. Q. Aguiar (Advisor)

Instituto Superior Técnico, University of Lisbon

Professor Takeo Kanade (Advisor)

Carnegie Mellon University

Abstract

We study the problem of 3D pose estimation of textureless shiny objects from monocular 2D images, for a bin-picking task. The main challenge of dealing with a shiny object comes from the fact that the object appearance largely changes with its pose and illumination. Therefore, conventional 3D-2D correspondence search usually fails due to the inconsistency of feature descriptors. For a textureless object like a mechanical part, visual feature matching becomes even harder due to the absence of stable texture features. Hierarchical template matching approaches require a larger number of templates to be matched when dealing with shiny objects, due to the drastic appearance changes with pose. In the challenging scenario of a bin-picking task, we must also cope with partial occlusions, shadows and inter-reflections, requiring redoubled effort in matching each template to obtain reliable results, which compromises the attractiveness of such approaches that are usually popular for textureless objects.

In this thesis, we develop a purely data-driven method to tackle the pose estimation problem. Motivated by photometric stereo, we develop an imaging system with multiple lights to acquire a multi-light image where channels are obtained by varying illumination directions. In an offline stage, we capture multi-light images of a given object in several poses. Then, we use random ferns to cluster the appearance of small patches of the multi-light images, and we store in each cluster the information of possible object poses. At run-time, the patches of the input multi-light image use the clusters information to probabilistically vote on several pose hypotheses. Since our pose hypotheses are a discrete set, we refine the discretized pose into the continuous space, in order to obtain accurate object poses for robotic manipulation.

Experiments show that the given method can detect and estimate poses of textureless and shiny objects accurately and robustly within half a second. We further compare our approach with the HALCON commercial software, a highly optimized hierarchical template matching approach developed by MVTec, and show some of the drawbacks of such type of approaches. Finally, we run detection on a different object by simply changing the image database.

Keywords: object pose estimation, monocular pose estimation, object recognition, object detection, random bin-picking, photometric stereo, probabilistic voting, random ferns, logistics, factory automation, manufacturing.

“We are like dwarfs on the shoulders of giants, so that we can see more than they, and things at a greater distance, not by virtue of any sharpness of sight on our part, or any physical distinction, but because we are carried high and raised up by their giant size.”

Bernard de Chartres

Acknowledgments

When I first applied to this dual Ph.D., little I knew that I was signing up for a travel through the world with deep-wide open eyes. To all the people that shared their inspiring journeys with me over these Ph.D. years, I owe my heartfelt gratitude.

This thesis has its genesis in my advisors João Xavier, Pedro Aguiar, and Takeo Kanade, who gave me the vision to pursue the big challenges in optimization, perception, and robotics. They let me "stand on their shoulders", giving me the support, the insight, and the knowledge that made this thesis possible. They also allowed me the unique opportunity of working with so many companies during my Ph.D.: Honda, Qualcomm, Google, and Mujin. Such opportunity, together with this dual Ph.D. program, gave me an invaluable chance to work with many diverse research institutions, teams, and mentors across many countries.

I am indebted to João Xavier and Pedro Aguiar, who first opened my eyes to the world of research and to my current expertise fields, who helped me reach to international conferences, and made with me my first trips around the globe. You did not only open many early doors for me, but you also walked me through the first steps of each very new path. Your scientific and life mentorship will surely remain fundamental building blocks of my critical thinking and reasoning. I am also thankful to Victor Barroso, Isabel Ribeiro, and João Paulo Costeira for their continuous support in those first steps of my scientific career.

I am honored to have known so closely Takeo Kanade, whom I crossed the ocean to look up to. You taught me by example how to look for the core ideas that solve a problem, how to take positively on them without restraint for what does not exist yet, and how to aim for perfection. I must thank you for your mentorship, and your perseverance in me. Your profound vision, simplicity, and honesty effortlessly made you my *Sensei*. I owe you for your guidance to the fields of bin-picking and robotic manipulation. They became the topic of this thesis, my fields of expertise, and, ultimately, my current way of contributing (giving back!) to the world.

I would like to thank my committee members Alexandre Bernardino, Jianbo Shi, and Marios Savvides, for giving valuable comments and criticisms that helped clear up the content, sharpen the final thesis message, and further outline future work. I also greatly thank you for your optimism and encouragement at the time of thesis defense.

During my Ph.D., I had the opportunity to be mentored, collaborate with, and discuss research topics with several people. I am thankful to Murali Chari and Sanjiv Kumar for their mentorship during my first internships, to Jun-Sik Kim and Rosen Diankov for doing great work together, to João Mota, Ricardo Cabral, Augusto Santos, Cláudia Soares, Dusan Jakovetić, Dragana Bajovic, Sérgio Pequito, and Pedro Guerreiro for walking this Ph.D. path together, along with many stimulating discussions, and finally to Ashwin Swaminathan, André Martins, and Mário Figueiredo, for putting the fields of Computer Vision and Machine Learning into a distinct perspective.

Many people along the way made sure my expatriate life was as much professional as personal, fun, and rich in experiences. For the friendships-at-first-sight, trips, get-togethers, long walks, and precious free-spirited company, I would like to thank Shubhra Kothari, Osmaan Akhtar, Nethra Krishnamoorthy, Natasha Kholgade, Kaushik Vaidyanathan, Anisha Nijhawan, José Caldeira, Lavanya Subramanian, Radhika Ramadoss, and Ramkumar Krishnan.

I would like to specially thank João Paulo Costeira for the friendship and devotion you have poured in the Carnegie Mellon | Portugal program and each one of us. The way you made our battles yours will be carried with us as an example of life. I surely owe you a great deal for entering the Carnegie Mellon | Portugal program and my successes in it. Your friendship goes back to the beginning of my graduate life and I hope will last for long.

I am deeply appreciative of the support by Carnegie Mellon | Portugal (ICTI) program and Portuguese Foundation for Science and Technology (FCT), without which this project would have not been possible. This work was sponsored under ISR/IST plurianual funding, through the PIDDAC Program funds, and Grants CMU-PT/SIA/0026/2009 and SFRH/BD/33521/2008. I would like to also thank the funding of Honda Engineering Co., Ltd, and the later support of the Ph.D. through BL/63/2014 of IST-ID.

To my parents Aníbal and Nazaré, who taught me the noble ways of living, and to my brothers and sisters, who raised up their little brother. You have been the ones that I looked up to in the early stages of my life. You have always encouraged me and helped me to move forward no matter which path I choose. I dedicate this thesis to all of you.

My final gratitude goes to my wife, Aditi. I thank you for all your support, kindness, and love. Despite the adventurous path I chose to take in these last years, you made yourself the pillar and compass of my life, making sure I finish the writing of this thesis, covering for me in parenting Zara, and accompanying me around the globe. Now I can only promise that the roller coaster will slow down for a brief period of time.

Contents

List of Tables

List of Figures

1

Introduction

1.1 Motivation

Industrial robots increasingly get deployed to serve *at scale* as factories and logistic centers become global in an e-commerce era. Such industrial sites must streamline and automate their processes in order to provide complex services in a swift way and compete worldwide.

While industrial settings are more than ever in need for automated solutions, the widespread adoption of robots is largely limited by their ability to manipulate objects in the surrounding environment. Many logistic and factory processes have been heavily automated in the past by designing constrained processes where robots can be taught to execute a small set of motions and little or no sensing is needed. However, many tasks in need of real-time complex 3D sensing and motion planning remain largely manual. One fundamental challenge, common to the vast majority of automated processes, must be solved to enable lights-out industrial sites: how to feed these automated processes with parts in a flexible and scalable manner, a task that remains hand-operated or tailored to particular scenarios. A key solution to such problem is known as "random bin-picking", where a robot manipulator senses, plans, and picks objects from an unstructured bin of parts into the automated process without any assistance of an operator.

Random bin-picking systems: an overview. Random bin-picking systems can be used in a wide range of applications, as depicted in Figure 1.1. Some of the most scalable, but very challenging, use-cases are:

- *Fulfillment of online retail orders.* The robot receives several stock-keeping units (bins), each with several instances of a customer-selected item, and fulfills the customer order by picking a number of items from each stock container into the costumer container that will carry the items for packing and shipping. Since each online order is tailored to a particular costumer, the robot must be able to handle a very large diversity of items.

- *Parts feeding for assembly lines and factory processes.* Several robots spread across the industrial site feed parts from bins into the automated processes. Contrarily to the robots fulfilling online retail orders, these robots are often required to pick the same few parts over and over again. However, as the manufacturing processes evolve and the final products adjust to the market needs, the same picking systems must be flexible enough to adjust to the changing needs of the industrial site.

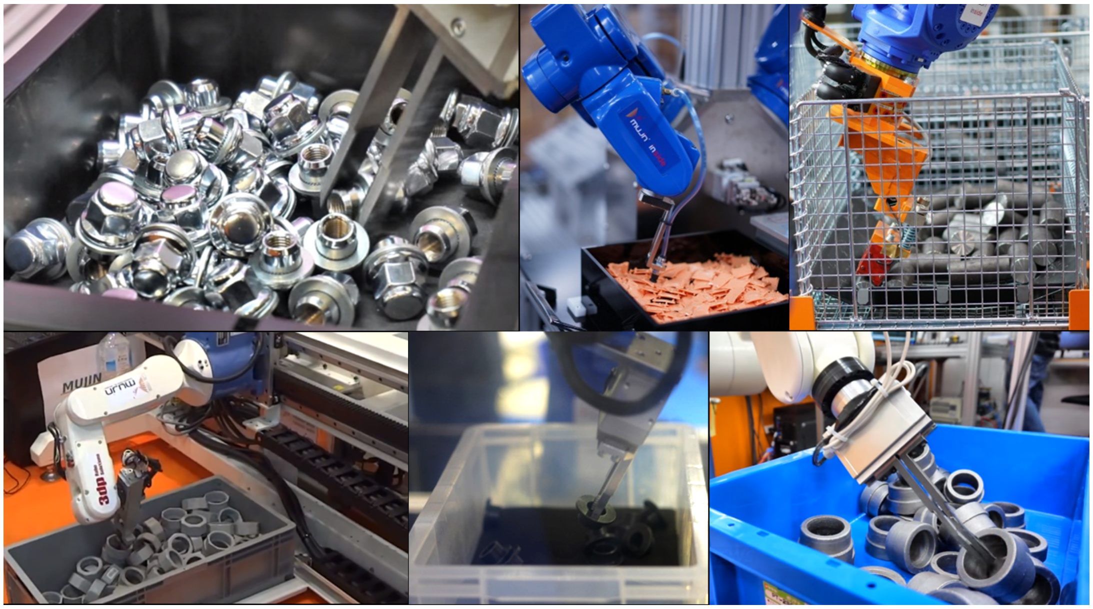


Figure 1.1: Several illustrations of bin-picking tasks in factory automation. Contrarily to 2D picking from a flat conveyor, picking from a deep bin involves real-time sensing and planning for completely arbitrary object poses. The scene is drastically cluttered with identical objects, hindering object segmentation from its background and promoting pose hallucinations. Images courtesy of Mujin, Inc.

In such applications, the robot must carefully pick parts from the bin. While one could be tempted to develop a system that would simply identify and aim for pickable regions, e.g., identifying planar regions for suction picking, such simplifications are largely unsatisfactory as robot planning tasks need detailed information of the object to manipulate. Motion planning needs full knowledge of the object pose in order to safely remove the part from the bin, avoid unstable grasps, take into account mass distribution, avoid collisions during the transfer and placement, and to make sure that the picking task is only initiated with hand grasps that allows the robot to place the part within the required placement pose constraints for the given task. Therefore, the robot must accurately detect and localize the parts in the three-dimensional space. For such a task, vision-based object detection and localization can be a cost-effective solution.

Object-specific registration and pose estimation. Within industrial sites, every part must be registered and carefully tracked. Either when fulfilling online retail orders or when feeding parts to a manufacturing process, the content of each storage bin is known and contains instances of only one registered part. Therefore, it is of major importance to focus efforts on solving the random bin-picking problem for bins that contain several instances of *one* given item.

Each object can be a-priori visually registered in order to teach the vision system the shape and appearance of the object to manipulate. This teaching process enables collecting accurately labeled training data, enabling for a plethora of visual and geometric feature statistics of the object surface to be computed, e.g., object geometry, key points and local descriptors, lines and curves, reflectance properties, properties of textures and micro-textures, and so on.

Pose estimation of shiny textureless objects. It is generally accepted that objects with abundant, stable, and distinctive local features from all possible viewpoints are easy to recognize and localize when shown alone to a camera in a non-cluttered background. While localizing these objects in the bin-picking scenario still present many challenges, mainly due to the difficulty of segmenting an object from its background with similar properties, the mere presence of such abundant, stable, and distinctive local features largely eases the pose estimation problem.

However, most objects lack the properties mentioned above, making vision-based methods face several major challenges in estimating the pose of industrial parts randomly placed in a bin:

- *Shiny surfaces.* Parts can be made of metal, involved in a tight plastic wrap, or simply made with a shiny finish that appeals to the buyer, thus frequently containing highly reflective surfaces. As a consequence, the object appearance becomes highly sensitive to the distribution of light, surface material, camera viewing direction, and object pose.

- *Lack of texture.* While some objects are fully textured, the large majority of objects are either fully textureless or abundant on textures on some viewpoints, e.g., viewpoints designed to be attractive to the potential buyer, but severely lacking texture in many other viewpoints. When dealing with such objects, detection and localization becomes even harder. The absence of rich texture frustrates the application of widely known feature descriptors [32], e.g., SIFT [3] or ORB [4].

- *Random pile.* The hallmark of bin-picking applications is having a scene cluttered with *identical* objects. This invalidates approaches like [13], that rely on distinguishing areas of interest from the background. Also, the abundance of identical objects in an image promotes the hallucination of (erroneous) object poses when observing local features without taking into account the adequate geometric constraints. Finally, the existence of occlusions and large appearance variations resulting from shadows and object inter-reflections further hinder pose estimation, refinement, and evaluation stages.

Figure 1.2 illustrates the drastic appearance changes in a bin-picking image full

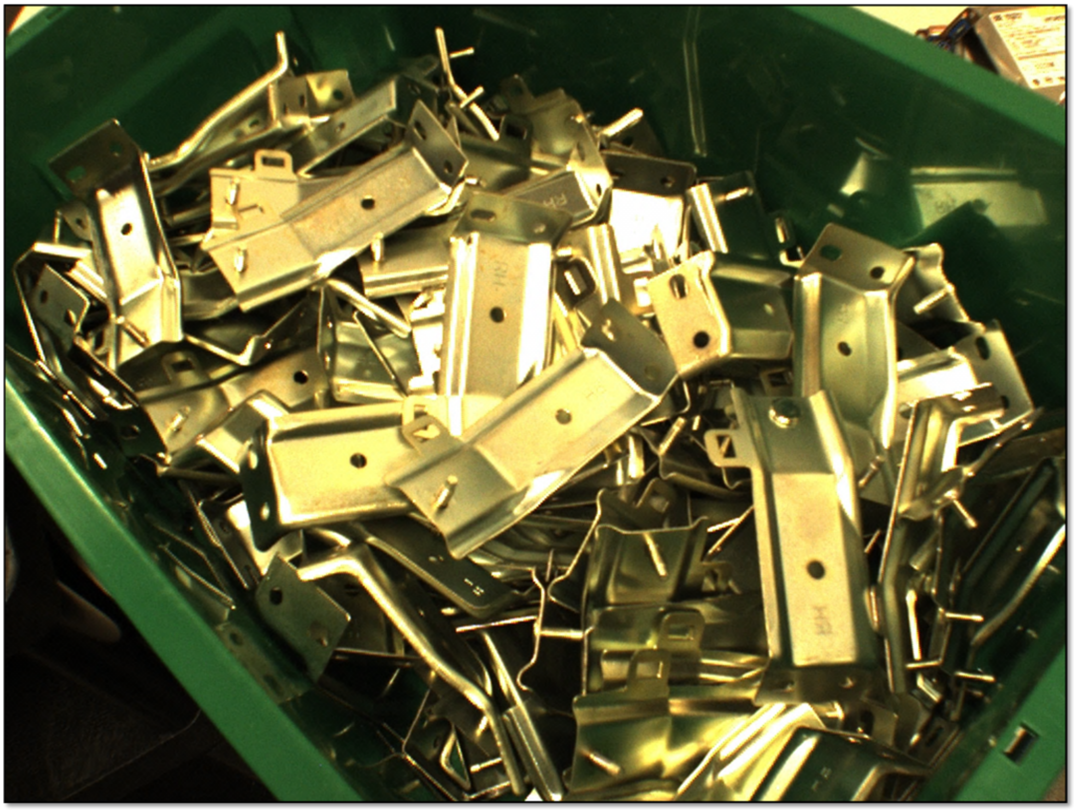


Figure 1.2: A typical bin-picking image of shiny objects. Detecting and localizing a shiny object from bin-picking images is challenging due to the high dynamic range of the image, sensitivity to small pose changes, occlusions, and large appearance variations descendant from shadows and inter-reflections.

of metallic shiny parts. The strong specular reflection in a narrow range of viewing directions widens the dynamic range of the image, and the intensity of the object surface becomes highly sensitive to pose change. Occlusions, shadows, and interreflections severely add to the instability of the object appearance.

Throughout this thesis, we focus on the localization of shiny textureless objects in a bin-picking scenario, from 2D images, showing our commitment in tackling the pose estimation problem in a setup that copes with all the problems mentioned above. While focusing on such specific most challenging scenario we believe that our discussion and solutions remain generic and can be applied to pose estimation of objects observed from both textured and textureless viewpoints, thus relevant to the pose estimation of a very wide class of objects.

1.2 Limitations of current approaches

Feature matching. One conventional way to estimate an object pose is to match visual features between an input image and a 3D object model [1, 2, 3]. In such a task, image features are matched to a dataset of features of the object in different poses using descriptors robust to viewpoint changes. For a shiny object, to tackle the discrepancy of appearance between different object poses, one needs to collect feature descriptors across a large set of object poses. Feature matching must then search for correspondences in a very large set of descriptors, often producing a substantial number of outliers. Additionally, visual feature matching works well for objects containing locally planar textures in all viewpoints [3, 5, 6, 7], which industrial metallic parts rarely have. Figure 1.3 depicts the difficulty in feature matching for shiny and textureless objects.

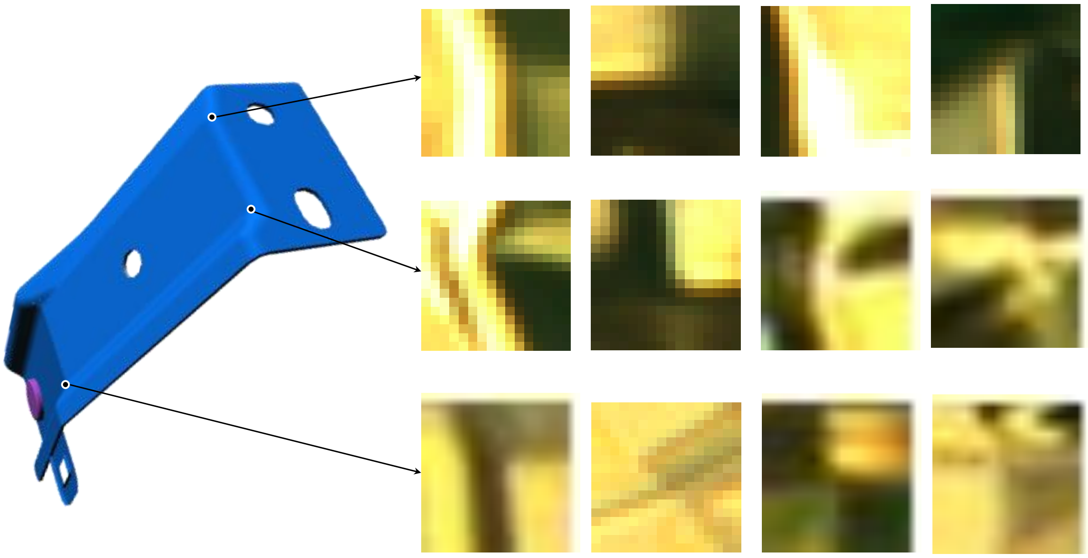


Figure 1.3: Appearance of imaged object points, under distinct object poses. Due to the textureless nature of the object, there is a scarce number of distinctive keypoints that could help localize the object under a feature matching framework. Due to the shininess of the object, the instability of the appearance of imaged object points further hinders feature matching.

Template matching. Another popular approach is to match whole templates [8, 9]. In this approach, synthetic or real images of an object in various poses are entered in a database. Given an input image, the object pose is estimated by searching the database for the most correlated example. Two major hindrances offset the approach simplicity: long computation time, due to the exhaustive search nature of the approach, and high sensitivity to appearance changes, due to occlusion, inter-reflection, and shadows. Recent works alleviate these two issues, but still fall short. The computation time can be significantly reduced via image pyramids and divide-and-conquer methods on a structured image database —as in the HALCON commercial software we compare against [8]. The appearance sensitivity can be mitigated using more robust appearance descriptors [10] and representing an image as a grid of patches [9]. However, the computation time and the accuracy of such methods depend on the number of templates matched at runtime and effort of matching each template. In bin-picking, the instability of the object appearance due to surface shininess, occlusions, shadows, and inter-reflections requires a redoubled effort in matching to keep accuracy high.

Specular cues. The use of specular cues —usually seen as nuisances [27, 28] —for pose estimation of shiny objects has been limited. The most popular approaches use the highlights from high-curvature surface points for feature matching, due to their robustness to pose changes [29, 31]. However, the availability of such specular cues depends on the object shape and pose, and is often scarce or nonexistent, thus restricting the applicability of the approach.

Contour matching. Object shape provides rich information for object identification and pose estimation. Several contour-based matching methods have been developed for cluttered background scenarios [24, 25, 26], but assume stable contours —a rarity in bin-picking conventional 2D images. Recent approaches use the shadows created with a multi-flash camera to detect depth edges [22], in an attempt to avoid the noise-sensitive clutter obtained with regular edge detectors. However, it is assumed that the shadows remain attached to the object, while, in practice, this depends on the shape of the object and the background. Also, illuminating a shiny object in only a narrow range of angles leads to a large set of surface orientations with dark appearance, difficulting contour detection.

Edge-based Hough voting. Several methods attempt to use the generalized Hough transform for detecting objects in the high dimensional pose space [11]. As in a conventional Hough voting, each edge point votes for poses. This is highly inefficient because the number of poses for each edge point is usually too large. In addition, it depends on the binarization, which is not so stable for a high-dynamic range image of a shiny object.

Patch-based Hough voting for 2D localization. Recently, similar voting approaches based on image patches, not on edge images, have been proposed using random forests for detecting and classifying objects [13, 17], and for matching interest points [5, 6] in a 2D image. Within our work, we use a voting procedure, similar to the ones used for 2D object localization in [13, 17], to hypothesize the 3D pose of textureless shiny objects in a bin-picking application. Instead of merely obtaining the centroid of objects in a 2D image, we perform full 3D pose estimation. In the latter, the ambiguity of voting for poses becomes much larger and the dimensionality of the voting space becomes a bottleneck in the localization procedure. This work is an extension of the work in [14], is patented [15], and is planned for submission in a journal [16].

3D Sensing for pose estimation. The current trend in vision systems for object pose estimation is to heavily rely on 3D point-clouds or RGB-D data to ease the pose estimation problem [9, 41, 42, 43, 44, 45, 51]. However, 3D data is often incomplete and with outliers in presence of shiny surfaces and inter-reflections while the pose estimation methods heavily rely on pointcloud or, worse, the corresponding very noisy normal map. In addition, high-quality 3D sensors remain expensive and slow. They are mostly based on structured light, slow due to the need of acquiring long sequences of patterns, or on active stereo sensing, faster in acquisition but significantly less accurate and computationally very intensive, using valuable time and system resources that could be otherwise used for pose estimation. Instead, we believe that, with re-doubled effort in monocular pose estimation, real-time weaker 3D sensing, e.g., time-of-flight (ToF) sensors, can become sufficient to complement 2D vision while remaining very cheap, computationally inexpensive, robust to ambient illumination, and real-time. Therefore we focus our work on object pose estimation from 2D images only, showing our commitment in improving pose estimation from 2D images rather than strongly depending on 3D data, while acknowledging the importance of 3D sensing to complement 2D vision as well as provide dynamic obstacle avoidance for reliable robotic manipulation in unstructured bin-picking.

1.3 Patch-based Hough voting for 3D localization

Our main goal is to design a practical 3D pose estimation system for bin-picking of shiny and textureless objects. We develop a multi-light imaging system coupled with a fully data-driven method to tackle the bin-picking problem.

Multi-light system. Inspired by photometric stereo, we develop a multi-light imaging system composed by a set of lights and a 2D camera, conceptually shown in Figure 1.4, where each channel of our multi-light image is acquired under variable illumination directions. Our controlled illumination setup ensures us a consistent surface appearance for a fixed scene. Our system is robust to external lightning conditions in the visible range as it is realized in the infra-red spectrum. The existence

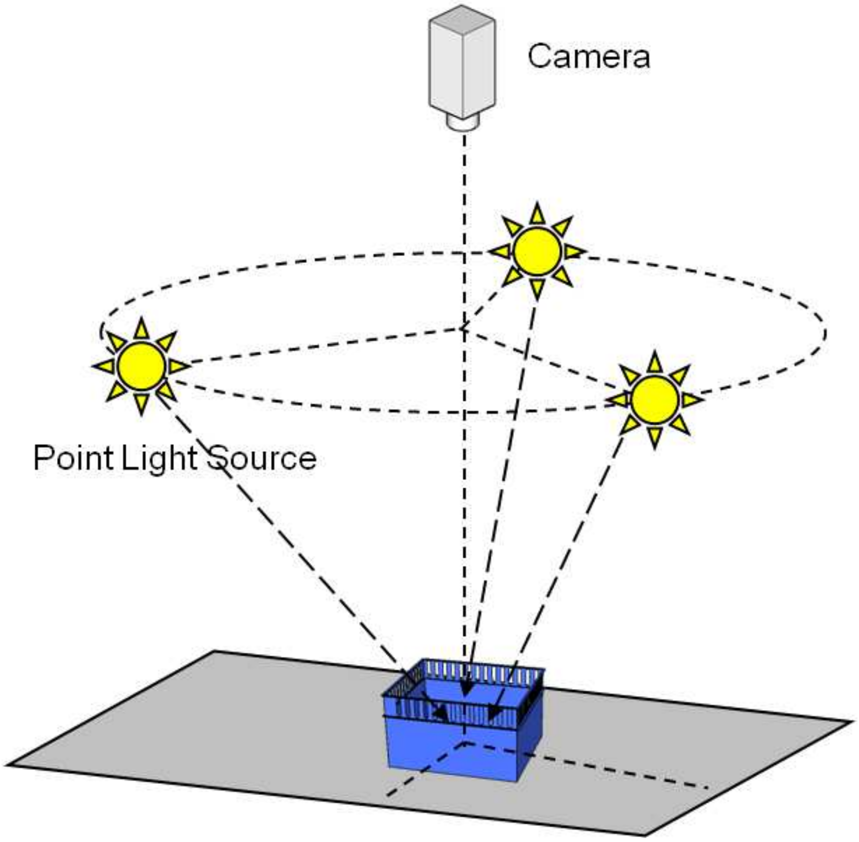


Figure 1.4: A conceptual diagram of our system using three light sources.

of multiple lights, each illuminating the scene from a different direction, creates images of objects with rich clues about the 3D pose of objects. Also, it attenuates the problem of having a large set of surface orientations exhibiting small radiance from the camera point of view.

Data-driven pose estimation. In order to deal with appearance changes without complicated modeling of the imaging process, we develop a purely data-driven approach using multi-light images to address the pose estimation problem:

Offline stage. At first, we capture multi-light images of a given object in several poses. Then, we densely collect small image patches, create an appearance codebook using random ferns and, for each codebook entry, we learn the probability of a pose given the appearance of an image patch. By using random ferns, the generation of the codebook becomes independent of the data and therefore computationally cheap, circumventing the large-sized optimization problem of clustering our huge amount of patches.

Online stage. At runtime, we infer object poses by gauging consensus among votes of image patches, using a new probabilistic voting framework for pose hypothesis generation. To the best of our knowledge there is no pose estimation approach describing pose hypothesis generation within a probabilistic framework from which generalized Hough Transform spontaneously emerge, thus benefiting from very strong outlier rejection. By using a voting-based approach, we narrow the pose search to a very small set of hypothesized poses, instead of exhaustively searching for poses as in template matching. We differ from 2D voting methods in that (a) we vote on the 3D pose space, (b) we present a probabilistic framework, contrarily to [13], and (c) we do not share the computational burden of iteratively refining the Hough votes as in the probabilistic approach in [17].

Voting optimizations. By searching for 3D poses, instead of simply localizing the object in 2D, the ambiguity of voting becomes much larger and the dimensionality of the pose space becomes a bottleneck in the voting procedure. To address these issues, we select the most discriminative patches for voting, in turn avoiding less informative patches that misspend time in voting for a large set of poses. In addition, we split the voting process into two steps, first searching in 2D for the object location and then searching for the 3D pose only at the most voted locations, avoiding a search over all possible poses. Finally, to mitigate the problem of discretization of the pose space and diminish the scattering of votes descendant from hypothesizing a pose from a small image patch, we gather consensus for a pose from votes for similar poses. In the 2D voting step, votes for neighbor 2D locations support each other. In the 3D pose search, we use sparse voting structures and an aspect graph to search for supporting votes, avoiding the computational burden of directly searching on the six degrees-of-freedom pose space.

Pose refinement. We upgrade our hypothesized poses, drawn from a discrete set, into the continuous space, in order to obtain accurate object poses for robotic manipulation. Towards this goal, we iterate the search for the best 3D pose from the hypothesized one by locally aligning the image edges with the object boundaries of the projected CAD model.

Experiments. Our results show that the given method can detect and estimate poses of textureless and shiny objects accurately and robustly within half a second. Our results also show that we can handle various objects by simply changing the image database, without a need for object-specific tuning of system parameters. We further compare our system with the HALCON software, a highly optimized commercial solution based on hierarchical template matching. Our system is faster, and yields a better recognition rate while providing comparable pose accuracy.

1.4 Contributions

The overall contribution of this thesis is a practical 3D pose estimation system capable of detecting poses of textureless and shiny objects for the bin-picking application. We highlight below the main features of our contribution:

- *Creating rich pose clues using a practical imaging system.* Our multi-light imaging system creates object images with rich pose clues, enabling us to explore the photometric properties of a textureless object to estimate its 3D pose. The system is inexpensive, easy to setup, and it works under unrestricted lighting conditions and unknown illumination directions.

- *Probabilistic voting for pose estimation.* Our method uses a new probabilistic voting framework that maps the appearance of multi-light image patches into votes for the object pose, allowing to bypass exhaustive search methods like template matching. Our approach is attractive due to the following reasons:

  - Our method can handle various objects by simply changing the image database, without a need for object-specific tuning of system parameters.
  - Our generation of pose hypotheses is fully data-driven, solving the appearance problem without complicated modeling of the imaging process.
  - We automatically select the most discriminative patches for voting, since they are the ones that carry rich pose information while taking less voting time.
  - Our 3D pose search first seeks for 2D objects centers and then performs a local 3D search per 2D center found, bypassing the memory and time issues of directly searching on the pose space.
  - We gather consensus from similar votes, mitigating the pose discretization problems and diminishing the effects of scattering of votes descendant from hypothesizing a pose from a small image patch.

  - Our approach makes use of random ferns, which do not require training to fix its binary questions, in order to circumvent the large-sized optimization problem of clustering our huge amount of patches at the training stage.
  - We use a new probabilistic framework that resembles the connection between voting frameworks and outlier-robust probabilistic models discussed in [50], from which —otherwise non-probabilistic —approaches related to the generalized Hough Transform spontaneously emerge.

- *Accurate pose estimation.* Our method refines the 3D pose hypotheses obtained in the voting procedure by using a visual servoing method, obtaining accurate object poses for robotic manipulation.

- *Benchmarking.* We compare our system with the HALCON software, a highly optimized commercial solution based on hierarchical template matching. Our system is more than five times faster and provides superior detection results with just a single fern.

1.5 Thesis organization

This thesis is organized in five chapters.

In Chapter 2, we describe our multi-light system. We compare its use with the traditional approaches of photometric stereo where these multi-light systems are mostly discussed. We also explain how we create our image database.

Chapter 3 discusses how we hypothesize object poses from the input images obtained with our setup. We first introduce a new probabilistic voting framework.

Then, using random ferns, we show how we map a patch appearance to a vote for an object pose. Finally, we describe how to refine the best pose hypotheses to obtain accurate object poses for robotic manipulation.

In Chapter 4 we illustrate the usefulness and robustness of our framework in the bin-picking application. We present our detection results and their accuracy, and we compare the performance of our approach with the performance of a commercial system. We also show detections on a different object by simply changing the image database.

Chapter 5 has some final remarks on the developed work.

In Chapter 6 we point out some of the limitations of our approach and discuss potential future developments that do not fall into the scope of this thesis but are possible extensions to the work presented.

2

Imaging with multiple lights

We use of an imaging system with multiple lights, as in photometric stereo [18], to obtain the pose of textureless shiny objects in a bin. Photometric stereo shows that imaging with such a system enables us to compute surface orientations, which is valuable information to determine the pose of objects. However, in order to compute surface orientations, photometric stereo requires an accurate model of the surface reflectance and distribution of lights. Hence, to achieve our goal of estimating object poses, we use the photometric stereo imaging system, but we do not compute the surface orientations of the observed scene or make any assumptions on the surface reflectance and distribution of lights. We rather infer object poses directly from images by using a fully data-driven approach.

In Section 2.1 we describe our multi-light system. In Section 2.2 we overview the photometric stereo method and its limitations. Section 2.3 describes how our multi-light system can be used for a data-driven pose estimation approach, discussing the advantages of bypassing the estimation of surface orientations.

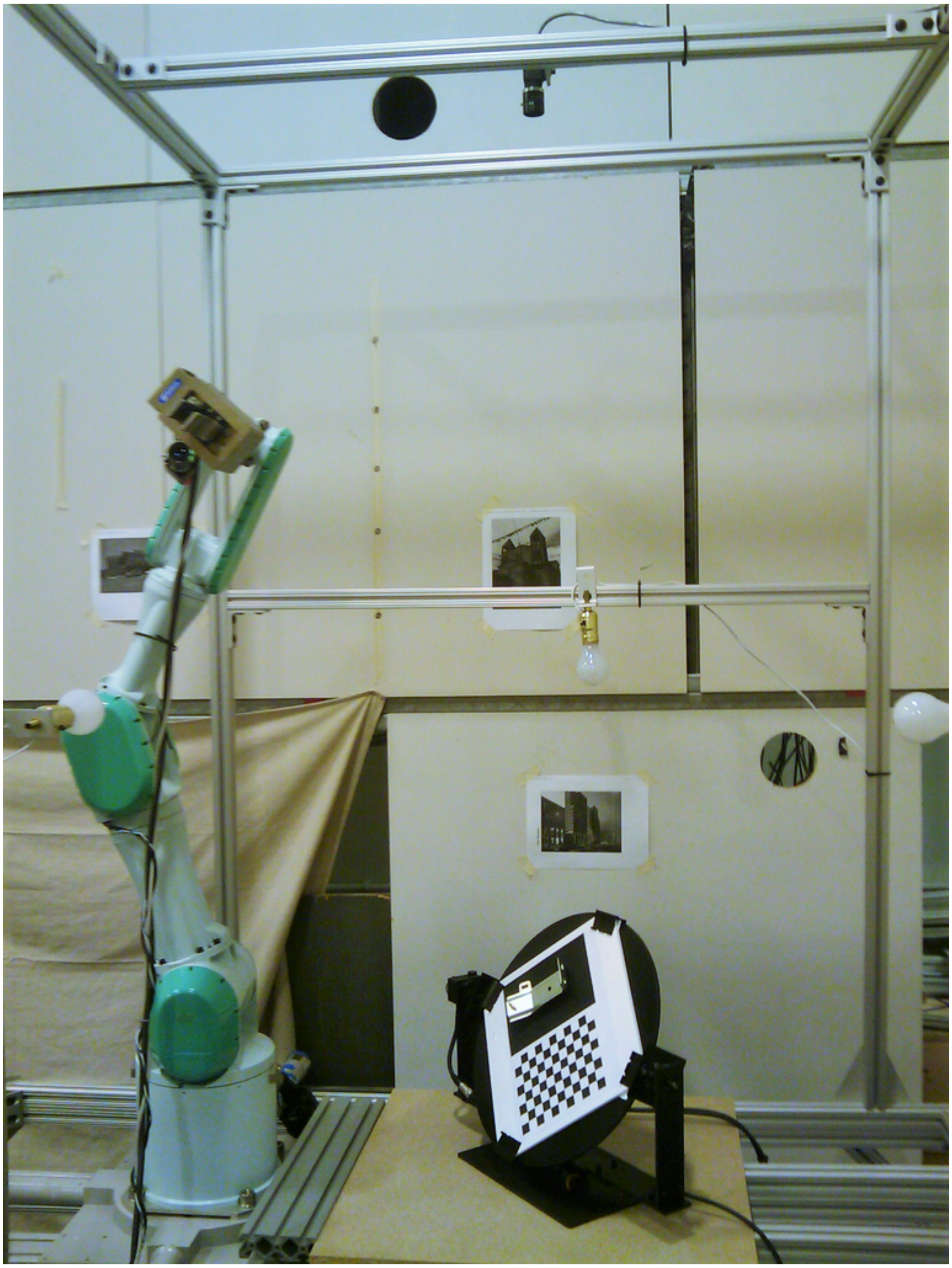


Figure 2.1: Real implementation of the multi-light source imaging system with a rotation stage for database collection. Figure 1.4 shows the corresponding conceptual system.

2.1 Multi-light images

Implemented imaging system. Our imaging system is the same as the one used in traditional photometric stereo. Figure 2.1 shows the implemented imaging system. It has three incandescent light bulbs, placed about 0.9 meter high from the bottom of the container and roughly located at vertices of a regular triangle. A B/W camera is about 1.75 meters high from the bottom of the container and aims to the center of the bin.

Multi-light images. With such a multi-light system, we acquire several grayscale images and arrange them in multiple channels of a single image. Since our system has three light sources, *we conveniently show the three grayscale images simultaneously in an RGB image* throughout this thesis, as shown in Figure 2.2. We call this RGB image a *multi-light image*, wherein the values of each pixel *encode* the orientation of the surface, a fact that we learn from photometric stereo. Such a multi-light image provides a high discrimination between different object poses.

Textureless shiny objects. In Figure 2.2, the images illuminated with a single light show large areas with small radiance. Such an effect occurs due to the reflectance property of a shiny surface illustrated in Figure 2.3. Shiny surfaces reflect a light ray in a narrow range of viewing directions, widening the dynamic range of the image. Our multi-light image mitigates this problem as it is acquired by illuminating the scene from distinct directions. Furthermore, our acquisition system creates color textures that provide rich clues about the 3D shape of the textureless scene.

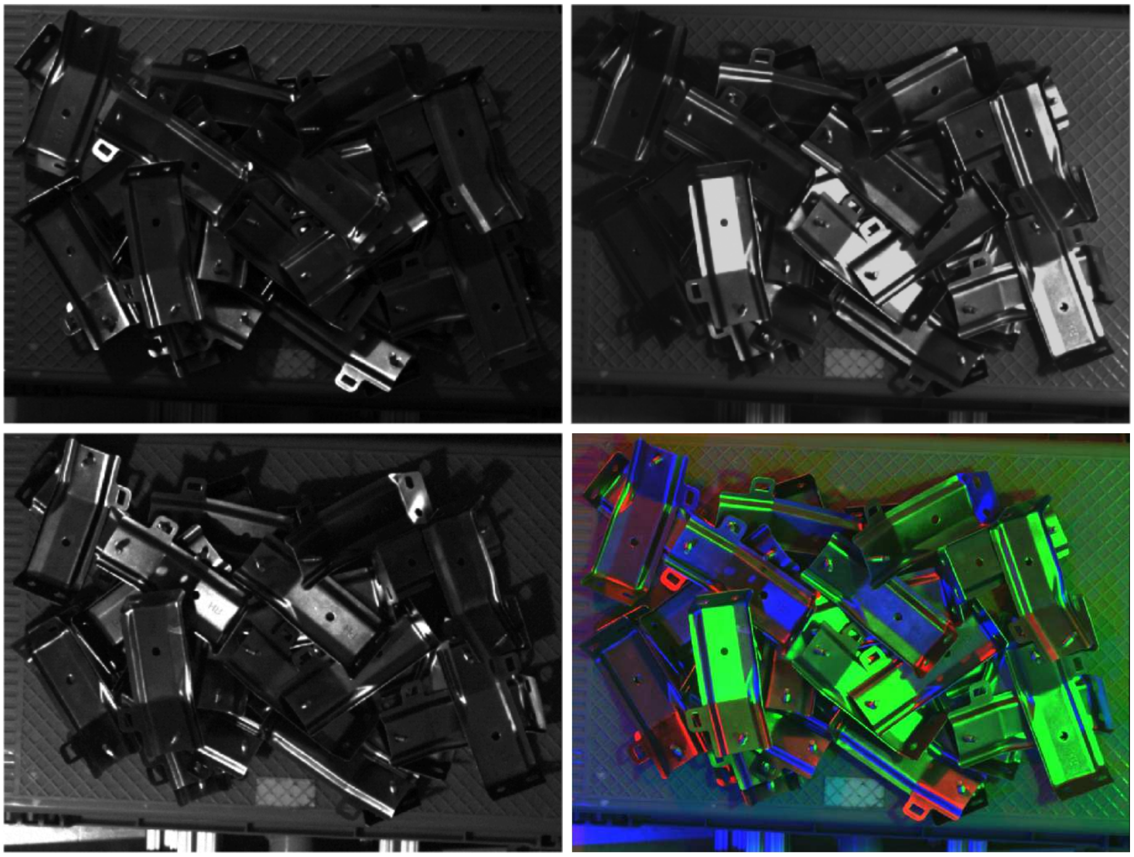


Figure 2.2: Multi-light image. Three grayscale images (top-left, top-right, and bottom-left) captured by the three light sources of the system in Figure 2.1. Through our document, we show them simultaneously in a single RGB image (bottom-right), one grayscale image per channel. While the images illuminated with a single light show large areas with small radiance, the colors in our multi-light image show rich clues about the 3D shape of the scene. (Best viewed in color)

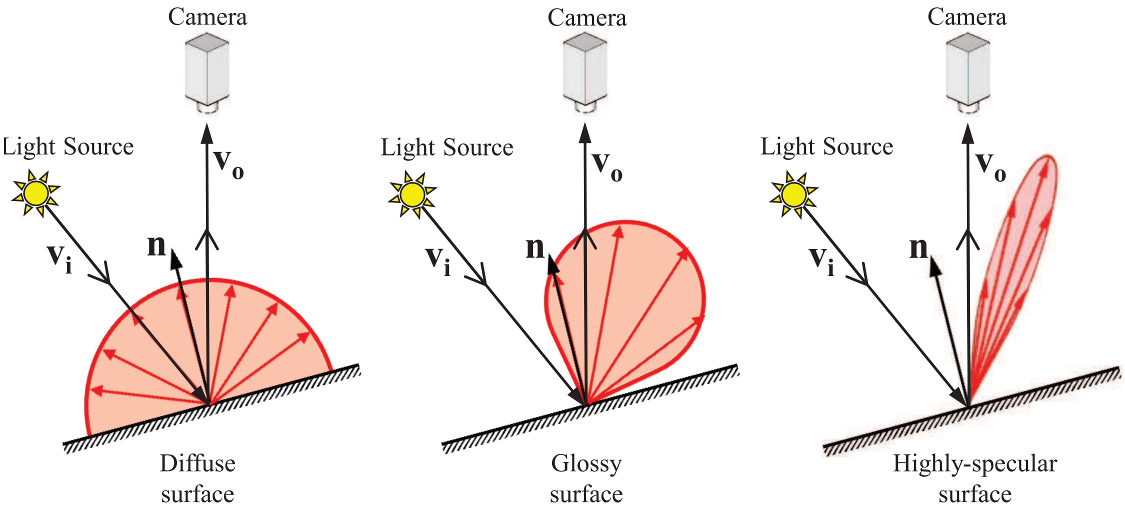


Figure 2.3: Reflectance models. While a purely diffuse surface emits the received light evenly in all directions, a more specular surface concentrates the reflected light within a narrow range of viewing directions.

2.2 Photometric stereo and its limitations

Photometric stereo. Photometric stereo [18] is a method for estimating the surface normals of a scene from its responses to multiple lights. Assuming a convex Lambertian surface, and non-existent ambient light, the intensity I of a point is related to its surface normal $\mathbf{n}$ and light direction $\mathbf{L}$ by

$$I = \rho \mathbf{n}^\top \mathbf{L}, \tag{2.1}$$

where ρ is the albedo of the point. For multiple light sources at known positions, the equation can be stacked

$$\begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} = \rho \mathbf{n}^\top \begin{bmatrix} \mathbf{L}_1 & \mathbf{L}_2 & \mathbf{L}_3 \end{bmatrix}, \tag{2.2}$$

and the surface normal $\mathbf{n}$ and its albedo ρ can be estimated by solving the linear system. Once the surface normal for every single point is estimated, one can try to reconstruct the 3D surface from the surface normals [33, 34].

Challenges with appearance. The above algorithm is based on some assumptions which are not appropriate for our case. The surface of shiny objects is not Lambertian, and the bin-picking scene or even the object itself is usually non-convex. Thus the imaging process can not be described as simply as in Equation (2.2). To model the imaging process for a shiny surface, even in the simplest scenario of homogeneous surface material and convex scene, an accurate bidirectional reflectance distribution function (BRDF) is required, which is not trivial to obtain. For non-homogeneous surface materials, in which the appearance may vary from point to point, and for a non-convex scene, where the appearance is often contaminated with shadows and inter-reflections, the complexity of the problem can dramatically increase.

Challenges with illumination. In addition, the above method assumes that all the light directions are known and that each intensity I is affected by a single light direction. In practice, this assumption is useful only when using point light sources or parallel lights with accurate system calibration, which is hard to achieve in a factory site.

From regular images to multi-light images. Equation (2.1) immensely increases in complexity for more realistic scenarios, e.g., with more complex direct and global illumination, non-lambertian surfaces with spatially varying reflectance properties, and non-convex objects and non-convex multi-object scenes. However, together with Equation 2.2, it states clearly why using multiple images acquired under distinct illumination directions greatly reduces the solutions provided by each equation from each image when estimating surface normals, a local cue of major importance for pose estimation. We believe that this principle remains true under the complexity of bin-picking images and therefore we use multi-light images in all our work, although bypassing the estimation of surface normals. We further provide experiments in Section 4.3 and Section 4.4 supporting the usage of multi-light images.

2.3 Using multi-light images for data-driven pose estimation

We use the images obtained with the multi-light system and, to overcome the limitations of photometric stereo, we follow a data-driven approach to determine object poses.

Data-driven approach. Using our acquisition system, we collect a large set of images of an object illuminated from different directions and observed from diverse viewpoints. We automatically label each of these images with the object pose using a checkerboard pattern as reference. The number of images in the database determines the resolution of the pose hypothesis. Typically, we capture about 16600 images of an object, obtaining an orientation resolution of approximately 4-degree while keeping the object position approximately constant. Then, as detailed in Chapter 3, we probabilistically learn to map the appearance of an image patch to 3D pose hypotheses of an object, exploiting the similarities between training and test images.

Dealing with appearance. Our data-driven pose estimation approach utilizes the discriminative photometric appearance among different poses without using complex imaging models and their calibration process. We naturally deal with non-homogeneous surfaces materials, inter-reflections from other surfaces of the same object, and unknown light distribution.

Since we do not restrict our lighting to parallel illumination, the observation angle of the light ray, the incident angle of a light ray, and the power of the light ray may vary with change in the object location, as illustrated in Figure 2.4. In such a case, the pixel intensities depend not only on the surface orientation but also on the location of the observed surface point. Thus, there exists a different correspondence between patch appearance and the object pose for each object location.

However, instead of learning a distinct map between patch appearance and pose hypotheses for every location, we learn a map for only one location. Our approach is robust to the appearance changes that result due to minor changes in the location, simplifying the learning procedure. While Figure 2.4 shows considerably changes in appearance per location, most of those changes can be mitigated by constructing a map invariant to local brightness changes, as discussed in Section 3.2. Our experiments show that only one map is enough to deal with all the appearance variations within the large volume of our bin.

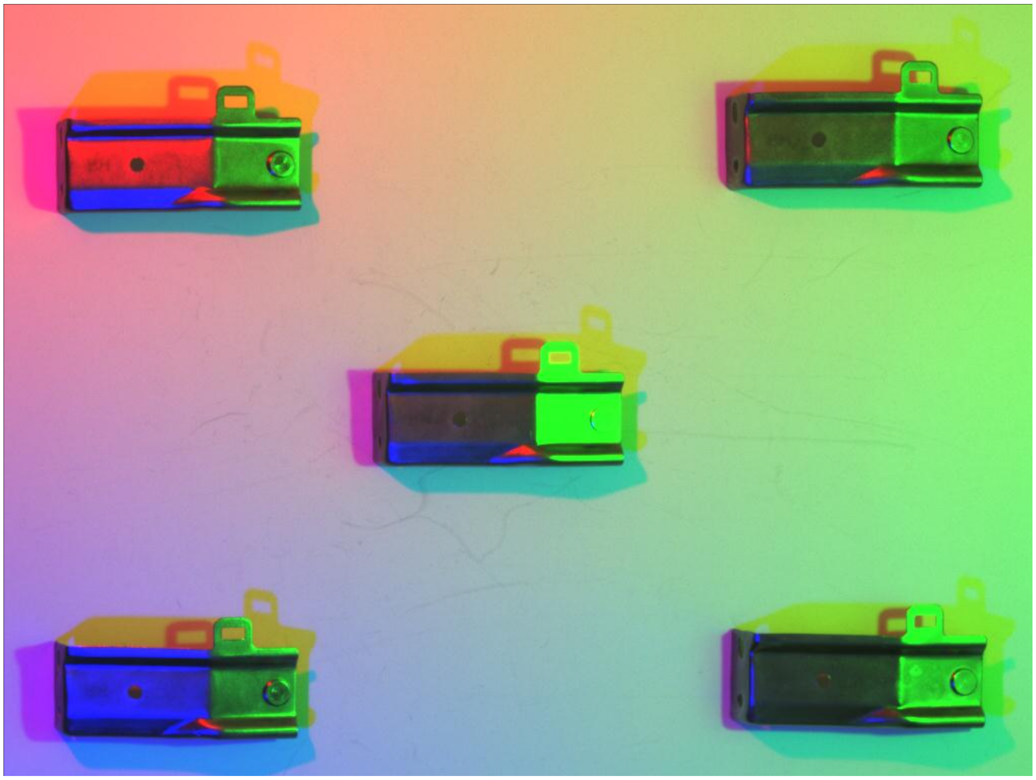


Figure 2.4: Appearance changes due to object translation. A shiny object was placed in distinct locations, while roughly maintaining its depth and orientation. The appearance of the shiny object changes significantly, specially when compared to the appearance of the diffuse background surface.

3

Data-driven pose estimation

In this chapter, we present a data-driven approach to tackle the pose estimation problem. We first introduce the probabilistic voting framework that we use to infer object poses by gauging consensus among patches. Then, we show how random ferns handle our huge amount of image patches and how they deal with the gargantuan number of possible patch appearances. We further discuss how we avoid the high memory and computational costs of directly searching on the six degrees-of-freedom pose space. We end this chapter describing how to refine the best pose hypotheses to obtain accurate poses.

3.1 Voting-based pose estimation

Consider a bin-picking scene where several objects are placed randomly in a bin, each with a distinct 3D pose with quantized (discrete) representation $\boldsymbol{p} = (\boldsymbol{x}, z, \boldsymbol{\theta})$, where $\boldsymbol{x} = (x, y)$ is the center of the object in image plane, z is the depth of the center of the object, and $\boldsymbol{\theta} = (\rho, \phi, \psi)$ is the object orientation. Our goal is to find the pose of the objects in the bin using 2D images taken by the system described in Chapter 2.

Probabilistic model. We represent a bin-picking image as a set of patches of size $n \times n$, sampled evenly on a grid and indexed by $k = 1, \cdots, K$. We designate a patch by $P_k = (\boldsymbol{x}_k, \boldsymbol{B}_k, \boldsymbol{Z}_k)$, where $\boldsymbol{x}_k$ is the patch center in the image plane, $\boldsymbol{B}_k$ its observed appearance, and $\boldsymbol{Z}_k$ is the latent (unknown) quantized pose of an object surface observed at an image point $\boldsymbol{x}_k$. For tractability of our model, we further assume that the random variables of each patch, $\boldsymbol{Z}_k$ and $\boldsymbol{B}_k$, are independent among patches, i.e., $(\boldsymbol{B}_k, \boldsymbol{Z}_k) \perp (\boldsymbol{B}_j, \boldsymbol{Z}_j), \forall_{j \neq k}$. Thus, we can write their joint probability as

$$\mathbb{P}(\boldsymbol{B}_1, \cdots, \boldsymbol{B}_K, \boldsymbol{Z}_1, \cdots, \boldsymbol{Z}_K) = \prod_{k=1}^{K} \mathbb{P}(\boldsymbol{B}_k, \boldsymbol{Z}_k). \tag{3.1}$$

Inference. Our goal is to infer the poses of the objects in a bin-picking image by observing the appearance of the image, which we break down into small image patches. One possible approach to the bin-picking problem would be to obtain the *maximum a posteriori* estimate

$$(\hat{z}_1, \cdots, \hat{z}_K) = \arg\max_{z_1, \cdots, z_K} \mathbb{P}(\boldsymbol{Z}_1 = z_1, \cdots, \boldsymbol{Z}_K = z_K | \boldsymbol{B}_1 = \boldsymbol{b}_1, \cdots, \boldsymbol{B}_K = \boldsymbol{b}_K). \tag{3.2}$$

which, using the independence assumption in Equation (3.1), becomes

$$\hat{z}_k = \arg\max_{z_k} \mathbb{P}(\boldsymbol{Z}_k = z_k | \boldsymbol{B}_k = \boldsymbol{b}_k), \ \forall_{k=1,\cdots,K}, \tag{3.3}$$

obtaining, for each patch position $\boldsymbol{x}_k$, the pose $\hat{z}_k$ of the visible object. However, a patch appearance $\boldsymbol{B}_k$ is usually not discriminative enough to accurately estimate the pose of the object observed at that location. More precisely, the conditional probability $\mathbb{P}(\boldsymbol{Z}_k | \boldsymbol{B}_k = \boldsymbol{b}_k)$ often has several equally probable maxima. In addition, with a gargantuan number of possible patch appearances $\boldsymbol{B}_k$ and poses $\boldsymbol{Z}_k$, it is hard to finely estimate every $\mathbb{P}(\boldsymbol{Z}_k|\boldsymbol{B}_k = \boldsymbol{b}_k)$,

Instead of estimating the poses $\boldsymbol{Z}_k$ at each position $\boldsymbol{x}_k$ independently, we turn to a random variable that gauges consensus among patches

$$\boldsymbol{W}_{\boldsymbol{p}} = \sum_{k=1}^{K} 1_{\boldsymbol{Z}_k = \boldsymbol{p}}, \tag{3.4}$$

where $1_{\boldsymbol{Z}_k=\boldsymbol{p}}$ is an indicator random variable that takes on the value 1 when $\boldsymbol{Z}_k = \boldsymbol{p}$ and 0 otherwise. The random variable $\boldsymbol{W}_{\boldsymbol{p}}$ counts the number of image patches exhibiting the quantized pose $\boldsymbol{p}$. Thus, $\boldsymbol{W}_{\boldsymbol{p}}$ is an Hough accumulator that aggregates pose information among patches, a capability that was lost in our probabilistic model when the independence assumption among patches was introduced.

Using the Hough accumulator $\boldsymbol{W}_{\boldsymbol{p}}$, we define a pose hypothesis $\hat{\boldsymbol{p}}$ as

$$\hat{\boldsymbol{p}} = \arg\max_{\boldsymbol{p}} \; \mathbb{E}(\boldsymbol{W}_{\boldsymbol{p}}|\boldsymbol{B}_1 = \boldsymbol{b}_1, \cdots, \boldsymbol{B}_K = \boldsymbol{b}_K). \tag{3.5}$$

Using our definition of $\boldsymbol{W}_{\boldsymbol{p}}$ in Equation (3.4), we first rewrite problem (3.5) as

$$\hat{\boldsymbol{p}} = \arg\max_{\boldsymbol{p}} \; \sum_{k=1}^{K} \mathbb{E}(1_{\boldsymbol{Z}_k=\boldsymbol{p}}|\boldsymbol{B}_1 = \boldsymbol{b}_1, \cdots, \boldsymbol{B}_K = \boldsymbol{b}_K). \tag{3.6}$$

Since, for an event A, $\mathbb{E}(\mathbf{1}_A) = \mathbb{P}(A)$, problem (3.6) becomes

$$\hat{\boldsymbol{p}} = \arg\max_{\boldsymbol{p}} \; \sum_{k=1}^{K} \mathbb{P}(\boldsymbol{Z}_k = \boldsymbol{p}|\boldsymbol{B}_1 = \boldsymbol{b}_1, \cdots, \boldsymbol{B}_K = \boldsymbol{b}_K). \tag{3.7}$$

Considering the independence assumption among patches present in our model, we obtain

$$\hat{\boldsymbol{p}} = \arg\max_{\boldsymbol{p}} \; \sum_{k=1}^{K} \mathbb{P}(\boldsymbol{Z}_k = \boldsymbol{p}|\boldsymbol{B}_k = \boldsymbol{b}_k). \tag{3.8}$$

As $\mathbb{P}(\boldsymbol{Z}_k|\boldsymbol{B}_k = \boldsymbol{b}_k)$ is hard to finely estimate, we replaced the outlier-sensitive pose estimate in (3.3) by the outlier-robust one in (3.8) that maximizes support among patches, resembling the probabilistic outlier models in [50].

Invariance to patch translation. Although (3.8) is already quite simplified, $\mathbb{P}(\boldsymbol{Z}_k|\boldsymbol{B}_k = \boldsymbol{b}_k)$ remains hard to finely estimate since we have a distinct probability function for every patch $\boldsymbol{P}_k$ in the image. More precisely,

$$f_{\boldsymbol{x}_k}(\boldsymbol{x}, z, \boldsymbol{\theta}; \boldsymbol{b}) := \mathbb{P}(\boldsymbol{Z}_k = (\boldsymbol{x}, z, \boldsymbol{\theta})|\boldsymbol{B}_k = \boldsymbol{b}) \tag{3.9}$$

depends on k (or $\boldsymbol{x}_k$ since $k \mapsto \boldsymbol{x}_k$ is a one-to-one map). In order to further simplify the function f in (3.9), we assume that the appearance of an object remains unchanged under any pose translation in $\boldsymbol{x}$. Under this assumption, a translation of a patch induces the same translation in the pose to estimate

$$f_{\boldsymbol{x}_k}(\boldsymbol{x}, z, \boldsymbol{\theta}; \boldsymbol{b}) = f_{\boldsymbol{x}_k+\delta}(\boldsymbol{x} + \delta, z, \boldsymbol{\theta}; \boldsymbol{b}). \tag{3.10}$$

Given the translation-invariance property in (3.10), there is a function g such that

$$f_{\boldsymbol{x}_k}(\boldsymbol{x}, z, \boldsymbol{\theta}; \boldsymbol{b}) = g(\boldsymbol{x} - \boldsymbol{x}_k, z, \boldsymbol{\theta}; \boldsymbol{b}), \forall_k. \tag{3.11}$$

We showed in (3.10) and (3.11) that the conditional probability distribution in (3.9) can be written as function of the offset $\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_k$, rather than depending on $\boldsymbol{x}$ and $\boldsymbol{x}_k$ separately. Thus, we create the random variables $\Delta\boldsymbol{Z}_{\boldsymbol{b}} \sim g(\boldsymbol{x} - \boldsymbol{x}_k, z, \boldsymbol{\theta}; \boldsymbol{b})$, only dependent on a patch appearance $\boldsymbol{b}$, and write problem (3.8) as

$$\hat{\boldsymbol{p}} = \arg\max_{\boldsymbol{p}} \sum_{k=1}^{K} \mathbb{P}(\Delta\boldsymbol{Z}_{\boldsymbol{b}_k} = \boldsymbol{p} - (\boldsymbol{x}_k, 0, \boldsymbol{0})). \tag{3.12}$$

where $\Delta\boldsymbol{Z}_{\boldsymbol{b}_k}$ is the pose transformation induced by the appearance $\boldsymbol{b}_k$ of patch $\boldsymbol{P}_k$.

Conceptually, under perspective effects and nonparallel illumination, the assumption that the appearance remains constant under pose translations is a good approximation only within a small volume, as illustrated in Figure 2.4. However, as explained in Section 2.3 and corroborated by our experiments, we show that a probabilistic function learnt at a single object location is enough to deal with all the appearance variations within the volume of our bin. This is because our binary representation of a patch appearance created from Equation (3.13) is invariant to local brightness changes. For larger scenes, the alternative to a larger multi-light system is to learn distinct probabilistic functions at a sparse set of object locations.

Implementation challenges. The alphabet of $\Delta \boldsymbol{Z}_{\boldsymbol{b}}$, for a given $\boldsymbol{b}$, is smaller than the one of $\boldsymbol{Z}_k$, since the offset $\Delta \boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_k$ is limited by the size of the object in an image. With such an alphabet reduction on the pose variable and a distribution $\mathbb{P}(\Delta \boldsymbol{Z}_{\boldsymbol{b}})$ shared across all k, we drastically diminished the complexity of the distributions to learn and consequently the amount of training data needed. Still, the gargantuan number of appearances $\boldsymbol{b}$ that a patch can have limits the applicability of (3.12), a problem that we tackle in the next section by quantizing the patch appearance into a visual codebook. In addition, implementing a voting algorithm on the pose space is time and memory consuming. Therefore, we describe in Section 3.3 how to speed-up the voting process by pose marginalization, a 2-step search algorithm that aims towards a more practical implementation.

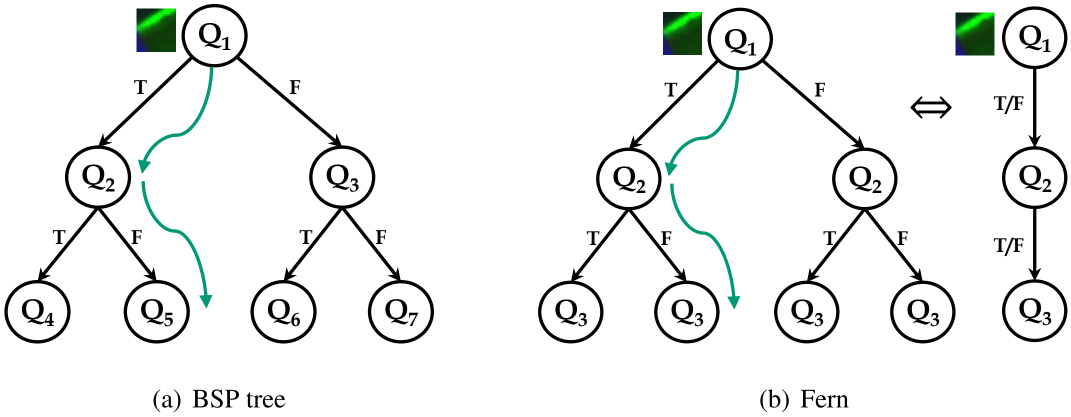
Revisiting voting approaches. In diverse work on object localization [13, 17], it is common practice to aggregate information from patches as a process of voting, either via a sum of probabilities or via sums of log-probabilities over all patches. Approaches performing sums of probabilities became widely popular because, contrarily to sums of log-probabilities, they do not excessively penalize scenarios which have insufficient data to finely learn the probability distributions needed. Equation (3.12) provides a probabilistic interpretation for some voting approaches based on sum of probabilities, opening doors for further improvements of such approaches within a probabilistic framework.

3.2 Visual codebooks using random ferns

In order to use the voting scheme in Equation (3.12), we now address the problem of effectively computing the probabilistic vote of a patch, since computing $\mathbb{P}(\Delta \boldsymbol{Z}_{\boldsymbol{b}})$ for all values of $\boldsymbol{b}$ is unfeasible. To compute the probabilistic vote of a given patch appearance $\boldsymbol{b}$, we search for similar patch appearances in our training database and extract their pose information. Below, we discuss the possible approaches for effectively extracting pose information from similar database patches.

Exhaustive search. Given an input patch at run-time, one possible approach is to search exhaustively for similar patches in the entire database, and then vote based on the pose information of the patches found. However, the number of patches in the database is huge, being approximately 130 million for our experiments. The dimension of $n \times n$ patches is $3n^2$ when using three light sources, which is also very large for our patch size $n = 17$.

Approximate nearest neighbor. Alternatively, we can use fast approximate nearest neighbor (NN) search methods to query our large database of patches. These methods usually use trees such as KD-trees, hierarchical k-means trees or ferns, where the querying time grows logarithmically with the database size. Using trees, a basic



(a) BSP tree

(b) Fern

Figure 3.1: Binary space partitioning (BSP) tree and a fern. The questions in the same level of a fern are the same, turning it into a non-hierarchical structure.

search for a NN candidate corresponds to traversing the tree and, upon reaching a leaf node, performing exhaustive search on the data points in that leaf. Such an approximate NN search requires all the data points to be in memory, which requires a large amount of memory for our database size. Also, the NN candidate might not contain useful pose information, e.g., due to the image noise, or might have multiple similar data points, from neighbor object poses, with useful information for voting. Instead of finding a single NN candidate and extracting its pose information, we extract the pose information from all the NN candidates in the leaf cluster, which does not require patch appearances to be stored. We explain below our choice of tree to implement such a procedure.

Random fern. We construct a visual codebook using binary trees. Given the large size of our database, we need a tree that is easy to train, consumes low memory, and has short retrieval time. For training a tree, optimally designing its questions requires solving large-sized optimization problems. In addition, for a binary tree with

$m-$levels, storing $2^m - 1$ questions for large trees consumes a large amount of memory. To simplify our training and test stages, we use a binary *random fern*, illustrated in Figure 3.1, which is a binary tree with only one question per level. Having only one question in each level of the $m-$level tree, independent of the ancestors, a fern becomes easily parallelizable. More importantly, there are only m questions to store in memory and access at run-time. The fern questions are designed via an easy random process, as described below.

Simple random binary questions. We use simple binary questions defined as

$$Q_i(\boldsymbol{b}) = \begin{cases} 1, & \text{if } \boldsymbol{b}(\boldsymbol{r}_i) - \boldsymbol{b}(\boldsymbol{r}'_i) < \tau_i \\ 0, & \text{otherwise} \end{cases} \tag{3.13}$$

at each level i of the fern, similar to the ones used in [6, 23]. Each question compares two intensity values in the patch appearance $\boldsymbol{b}$, at locations $\boldsymbol{r}_i$ and $\boldsymbol{r}'_i$, with a threshold τ_i. Since the image has multiple channels, the location $\boldsymbol{r} = (x, y, c)$ in the patch includes the channel c as well.

The number of possible pixel comparisons for $n \times n$ three-light image patches is $3n^2 \times 3n^2$, which makes the design of the m questions a large optimization problem. We circumvent such a large-scale optimization problem by alternating among channels and choosing two random points in the same channel and a random threshold value.

Storing pose information at leaves. In the training stage, we assign to each patch a label l composed by the answers to the m questions of the fern. Then, each leaf of the fern collects the pose information $(\Delta\boldsymbol{x}, z, \boldsymbol{\theta})$ of the patches with the same label l. For each label l, with a set S_l of votes $\boldsymbol{v}_j = (\Delta\boldsymbol{x}_j, z_j, \boldsymbol{\theta}_j)$, $j = 1, \cdots, |S_l|$, we

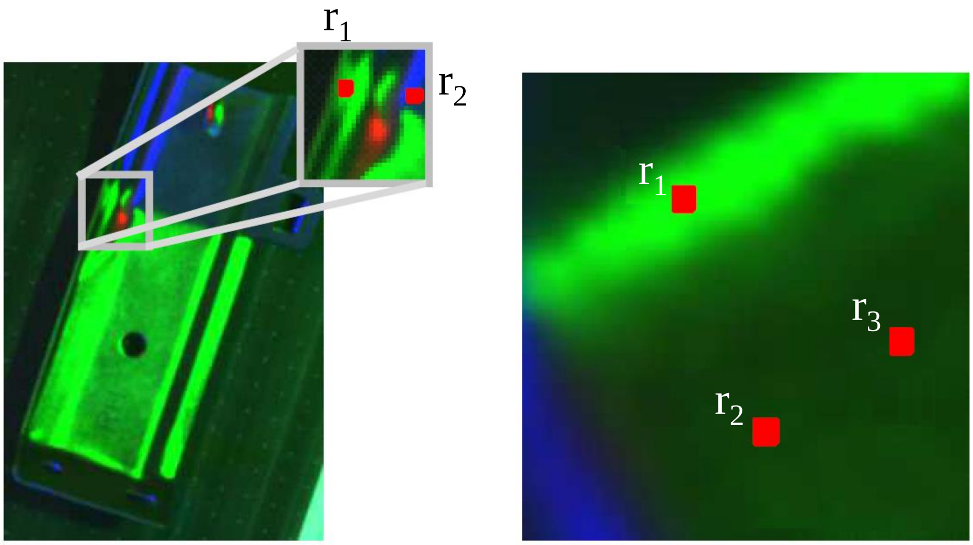


Figure 3.2: Binary questions in a fern. For $\tau = 0$, the question between locations $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ is informative for the given patch, but one between $\boldsymbol{r}_2$ and $\boldsymbol{r}_3$ is determined randomly by the camera noise. (Best viewed in color)

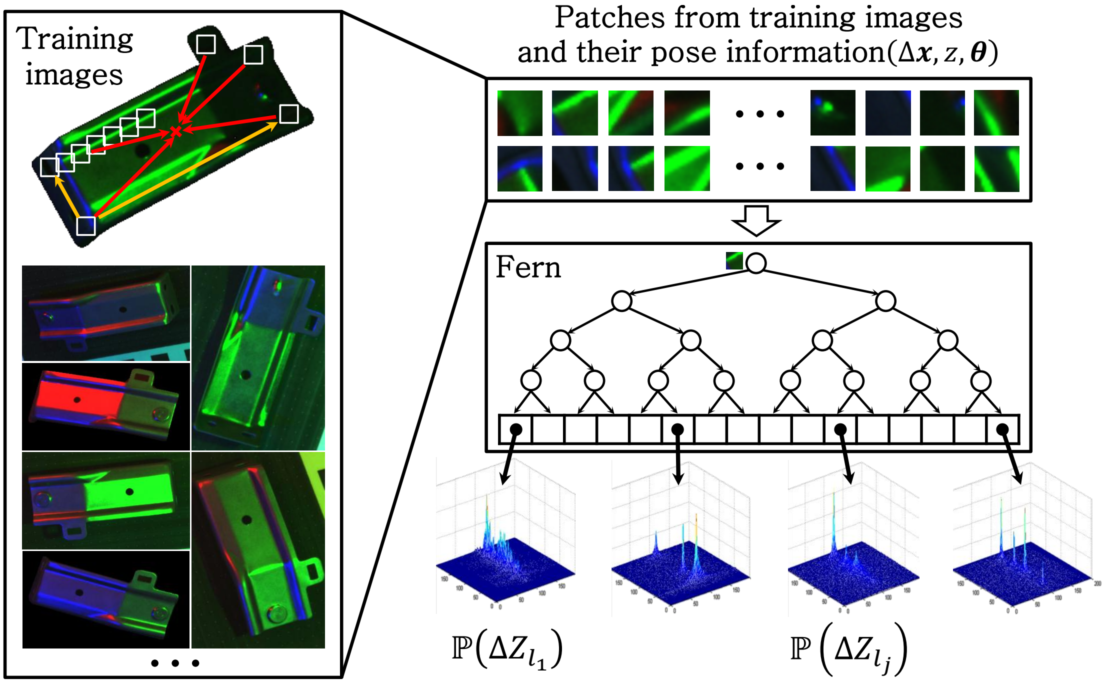

Training images
Patches from training images and their pose information$(\Delta \boldsymbol{x}, z, \boldsymbol{\theta})$
Fern
$\mathbb{P}(\Delta Z_{l_1})$
$\mathbb{P}(\Delta Z_{l_j})$

Figure 3.3: Illustration of the training procedure. We compute $\mathbb{P}(\Delta \boldsymbol{Z}_l)$ at each leaf of the fern, which will be used at run-time as a probabilistic vote.

compute the voting probability as

$$\mathbb{P}(\Delta \boldsymbol{Z}_l = (\Delta \boldsymbol{x}, z, \boldsymbol{\theta})) = \frac{\sum_{j=1}^{S_l} 1_{\Delta \boldsymbol{Z}_l = \boldsymbol{v}_j}}{|S_l|}, \tag{3.14}$$

where $|S_l|$ is the cardinality of S_l and $1_{\Delta \boldsymbol{Z}_l = \boldsymbol{v}_j}$ is an indicator random variable that takes on the value 1 when $\Delta \boldsymbol{Z}_l = \boldsymbol{v}_j$ and 0 otherwise. Figure 3.3 illustrates the training procedure.

At run-time, we use this probability distribution to cast votes for poses given a label l computed from each patch appearance $\boldsymbol{b}$ in the input image. It is much more feasible to compute $\mathbb{P}(\Delta \boldsymbol{Z}_l)$ for every label than computing $\mathbb{P}(\Delta \boldsymbol{Z}_{\boldsymbol{b}})$ for all patch appearances.

Depth of the fern. For about 130 million patches in our database, we use 27 questions in the fern, which results in 2^{27} (over 134 million) leaves for nearly the same number of training patches. We are deliberately over-splitting the patch space, leaving about 90% of the 2^{27} leaves empty. As a result, at run-time, a patch label may have no corresponding training data. This allows us to detect test patches that are clearly distinct from the trained ones, e.g., due to shadows or inter-reflections, and prevent them from voting for a pose, considering the difficulty of retrieving relevant pose data from their appearance. On the contrary, under such over-splitting, having a multitude of training patches in a single leaf indicates that such a patch appearance is too common to be used in the voting process and can be easily discarded. Thus, we disregard the fern leafs with more than 100 patches, therefore using only the most discriminative patches for voting while avoiding large clusters that substantially increase the voting time while adding low certainty about the object pose.

Dealing with multiple channels. The multi-light image captured by our system in Figure 2.1 implicitly encodes the surface normal information in the form of color. Since the light sources in our system are not far enough to provide parallel illumination, they often produce significant variations in the image color while changing the surface location and maintaining the orientation, as shown in 2.4. Therefore, there is no unique map between color and surface orientation for all image locations, which restrains the use of a single fern to map the patch appearance to object poses. The answers to the fern questions comparing values of a pixel across multiple channels, i.e., reasoning on the color of a single pixel, are not robust to surface translation. On the contrary, the comparison of pixels values within the same channel of a patch is robust to surface translation. Thus, in Equation (3.13), we define each question in one channel at a time, and not across multiple channels. This allows us to create a single fern useful on the whole volume of the bin of objects, drastically diminishing the number of images to be acquired and simplifying the training and online procedure.

In case the light sources are far enough to be assumed parallel, or the database image is sampled over a dense grid of (x, y, z) locations, we can use the assumption that the color is strongly related to the surface orientation. In this case, it would make more sense to design questions across multiple channels. Still, such questions would be sensitive to inter-reflections, since a channel brightness can change considerably under inter-reflections. In contrast, our current questions are robust to such brightness changes, and inter-reflections effects are often nearly constant within a neighborhood.

3.3 Online algorithm for pose hypotheses generation

The online algorithm consists of aggregating the votes from each patch to generate several pose hypotheses. For each patch in the input image, we obtain its label l by asking the m questions of the fern. Then, from leaf l, we retrieve the set S_l of votes $\left\{(\Delta\boldsymbol{x}_j, z_j, \boldsymbol{\theta}_j)\right\}_{j=1}^{|S_l|}$, with cardinality $|S_l|$. By accumulating all the votes from all the patches in the input image and finding the highest voted poses, we generate reliable pose hypotheses. As in the training stage, homogeneous patches do not participate in this online voting process, since they are usually non-discriminative about the object pose.

Algorithm 1 summarizes our approach at the online stage. In the sequel, we discuss some of the strategies for speed and robustness of the given algorithm.

```
Algorithm 1: Pose hypothesizing algorithm
Given a set $S_l = \left\{(\Delta\boldsymbol{x}_j, z_j, \boldsymbol{\theta}_j)\right\}_{j=1}^{|S_l|}$ of votes for each label $l$, with cardinality
$|S_l|$, and $m$ random fern questions fixed at training stage:
1. Compute the gradient of the input image.
2. Choose pixels $\boldsymbol{x} = (x, y)$ with large image gradients.
3. Allocate memory for the 2D voting accumulator $V_I(\boldsymbol{x})$ and the sparse
   accumulator for 3D pose votes $V_{z,\boldsymbol{\theta}}(\boldsymbol{x})$.
4. for each pixel $\boldsymbol{x}_k$ with large image gradients, vote with $\mathbb{P}(\Delta\boldsymbol{Z}_{l_k} = (\Delta\boldsymbol{x}_k, z, \boldsymbol{\theta}))$
   given by Equation (3.14), i.e, do
    4.1 Compute the label $l_k$ of the image patch centered at $\boldsymbol{x}_k$ by asking the
        $m$ fern questions.
    4.2 Retrieve the set $S_{l_k}$ of votes $\left\{(\Delta\boldsymbol{x}_j, z_j, \boldsymbol{\theta}_j)\right\}_{j=1}^{|S_{l_k}|}$ of the label $l_k$.
    4.3 for each vote $(\Delta\boldsymbol{x}_j, z_j, \boldsymbol{\theta}_j)$ do
        Add the voting confidence $1/|S_l|$ to $V_I(\boldsymbol{x}_k + \Delta\boldsymbol{x}_j)$.
        Insert the pose information $(z_j, \boldsymbol{\theta}_j)$ and its confidence $1/|S_l|$ in
          $V_{z_j,\boldsymbol{\theta}_j}(\boldsymbol{x}_k + \Delta\boldsymbol{x}_j)$.
5. Apply a Gaussian Parzen window to $V_I(\boldsymbol{x})$, to gather confidences for each
   location from neighboring locations.
6. Search for object locations $\hat{\boldsymbol{x}}$ in $V_I(\boldsymbol{x})$ with the highest voting score.
7. For each peak $\hat{\boldsymbol{x}}$, retrieve votes from all $V_{z,\boldsymbol{\theta}}(\hat{\boldsymbol{x}})$.
8. Gather voting confidences for a pose $(\hat{\boldsymbol{x}}, z, \boldsymbol{\theta})$ from neighboring poses.
9. Search for the pose hypotheses among $(\hat{\boldsymbol{x}}, z, \boldsymbol{\theta})$ with the highest voting
   score.
```

Speeding-up by pose marginalization. We implement the result obtained in Equation (3.5-3.12) using random ferns. The online algorithm becomes

$$\begin{aligned} V(\boldsymbol{x}, z, \boldsymbol{\theta}) &= \mathbb{E}(\boldsymbol{W}_{\boldsymbol{x},z,\boldsymbol{\theta}} | \boldsymbol{B}_1 = \boldsymbol{b}_1, \cdots, \boldsymbol{B}_K = \boldsymbol{b}_K) \\ &= \sum_{k=1}^{K} \mathbb{P}(\Delta \boldsymbol{Z}_{l_k} = (\boldsymbol{x} - \boldsymbol{x}_k, z, \boldsymbol{\theta})). \end{aligned} \tag{3.15}$$

Though the online algorithm is simple, it requires a huge accumulator of $(\boldsymbol{x}, z, \boldsymbol{\theta})$. For example, if the image size is 1024 × 768 and the number of database images, i.e., the number of possible depths z and orientations $\boldsymbol{\theta}$, is 16600, the number of accumulator bins is more than 13 billion. Searching for pose hypotheses in this huge accumulator is impractical. Thus, to accelerate the search without decreasing our search resolution, we provide an approximation to Equation (3.15) by splitting the search for poses into a 2-step search method.

In the initial voting stage, a two-dimensional voting accumulator $V_I(\boldsymbol{x})$ for object centroids $\boldsymbol{x}$ is considered by marginalizing the remaining pose dimensions $(z, \boldsymbol{\theta})$, thus evaluating

$$\begin{aligned} V_I(\boldsymbol{x}) &= \sum_{z,\boldsymbol{\theta}} V(\boldsymbol{x}, z, \boldsymbol{\theta}) \\ &= \sum_{z,\boldsymbol{\theta}} \sum_{k=1}^{K} \mathbb{P}(\Delta \boldsymbol{Z}_{l_k} = (\boldsymbol{x} - \boldsymbol{x}_k, z, \boldsymbol{\theta})). \end{aligned} \tag{3.16}$$

This is simply achievable by labeling each patch and accumulating its votes in the accumulator independently, as shown in Algorithm 1. Peaks $\hat{\boldsymbol{x}}$ in $V_I(\boldsymbol{x})$ are likely to have the best pose hypotheses $(\hat{\boldsymbol{x}}, \hat{z}, \hat{\boldsymbol{\theta}})$ since wrong votes usually scatter randomly.

Once the peaks $\hat{\boldsymbol{x}}$ in the 2D accumulator $V_I(\boldsymbol{x})$ are picked up, we search for the best pose hypotheses in $V(\boldsymbol{x}, z, \boldsymbol{\theta})$ only for the selected peaks, which is a search within just a few sparse votes. For this search, we have a sparse accumulator $V_{z,\boldsymbol{\theta}}(\boldsymbol{x})$

that contains the votes for each $(z, \boldsymbol{\theta})$ at each pixel $\boldsymbol{x}$. Because only a few poses are voted for each pixel, $V_{z,\boldsymbol{\theta}}(\boldsymbol{x})$ is an array of lists containing poses and votes for efficient memory usage and faster search.

Since the number of bins visited in the two-step search process is greatly reduced compared to the original search in the 6-DOF pose space, generating pose hypotheses becomes much faster.

Vote support from neighbor poses. Patches from the same object in the input image tend to scatter votes around similar poses, instead of consistently voting for the same pose. Such scattering occurs because objects in a slightly different pose have similar appearance, making it hard to accurately estimate a pose from the appearance of a small patch. Moreover, the observed object pose often differs from the quantized pose hypotheses, forcing the patches to vote for neighbor poses. To improve the robustness to errors due to pose quantization and difficulty of hypothesizing a pose from a small image patch, we boost the voting score of a pose by taking into account the votes for neighbor poses.

When computing $V_I(\boldsymbol{x})$, given a patch at $\boldsymbol{x}_k$, we alter the voting confidence $1/|S_l|$ of each pose hypothesis $(\Delta\boldsymbol{x}_j, z_j, \boldsymbol{\theta}_j)$ in a leaf l to

$$\frac{1}{|S_l|}\frac{1}{2\pi\sigma^2}exp\left(-\frac{\left\|(\boldsymbol{x}-\boldsymbol{x}_k)-\Delta\boldsymbol{x}_j\right\|}{2\sigma^2}\right). \tag{3.17}$$

where $\sigma^2\boldsymbol{I_{2\times2}}$ is the covariance of the Gaussian Parzen window. This can be efficiently computed by Gaussian-filtering the voting confidences $1/|S_l|$ accumulated at each pixel of $V_I(\boldsymbol{x})$.

In the second step of our voting procedure, we must search for neighbor poses around peak $\hat{\boldsymbol{x}}$ in $V_I(\boldsymbol{x})$. We first collect sparse sets of pose hypotheses from $V_{z,\boldsymbol{\theta}}(\boldsymbol{x})$

at neighbors $\boldsymbol{x}$ of $\hat{\boldsymbol{x}}$. Then, we efficiently identify neighbor poses using an aspect graph where each training pose has a precomputed list of neighbors. Finally, for every voted pose $(\hat{\boldsymbol{x}}, z, \boldsymbol{\theta})$, we gather votes from its neighbor poses, weighted equally in all the reported experiments. A possible improvement can include a weight factor representing the pose dissimilarity.

Robustness of voting to noisy labeling. Since images are contaminated by noise, aggravated by infra-red imaging, similar training patches are likely to scatter to different fern labels, in turn dispersing the information that is used to build a probabilistic vote. Consequently, when a test patch gets labeled, such a label frequently contains information from only a small subset of similar patches in the database, thereby weakening the vote quality. Also, test patches are often corrupted by shadows and inter-reflections, which make them substantially different from training data. We counterbalance such issues of ANN patch search with our voting mechanism. Since wrong votes usually scatter randomly and the voting space is very large, the probability of accumulating several votes for the same pose with random wrong votes is small. By using an ensemble of random forests, instead of a single random fern, we can make our voting mechanism even more robust, a topic whose details we leave for future work but has shown to improve object recognition and pose estimation results [6]. The number of decision trees to use in a forest is a tradeoff between accuracy, memory usage, and run-time. We expect the labeling and voting time of 136ms, reported in Table 4.4, to increase linearly with the number of ferns used, while keeping the remaining processing times nearly constant.

Memory usage. As previously discussed, we avoid the use of a large voting accumulator via a 2-step pose search. In addition, we tune our training data in several ways for a small memory footprint at the online stage. First, we do not store the appearances or any descriptors of the training patches in the fern leaves. Second, the number of possible object depths and orientations $(z, \boldsymbol{\theta})$ considered for voting is limited by the number of images taken in the training stage. Thus, the pose information $(\Delta\boldsymbol{x}, z, \boldsymbol{\theta})$ in a fern can be simply represented by $(\Delta\boldsymbol{x}, \boldsymbol{i})$, where $\boldsymbol{i}$ is the training image index corresponding to $(z, \boldsymbol{\theta})$, reducing our memory usage to two 16-bit long integers for $\Delta\boldsymbol{x}$ and one unsigned 16-bit long integer for $\boldsymbol{i}$, a total of 5 bytes per training patch. Third, in our implementation we do not store leaves with more than 100 patches, as the patches with such label are considered non-discriminative to vote for the object pose, spending too many resources for the value added by their votes. Finally, we do not store the individual voting confidences from each training patch. Instead, we just store the number of training patches $|S_l|$ with label l, from which the voting confidences $1/|S_l|$ can be computed and used in Algorithm 1, which can be represented by an unsigned 8-bit long integer as we limit it to 100.

3.4 Pose refinement

Each generated pose hypothesis $(\hat{\boldsymbol{x}}, \hat{z}, \hat{\boldsymbol{\theta}})$ is in a discretized space.

As illustrated in Figure 3.4, the pose hypothesis is not accurate because the pose of the object in the input image is usually not the same as any of the discretized pose hypotheses. A denser sampling of the training poses may alleviate this problem, but can not solve it. We describe how to upgrade the discretized pose into the 3D continuous space, and our criteria for rejection of wrong pose hypotheses.

Refinement procedure. To estimate a more accurate 3D pose, we refine the

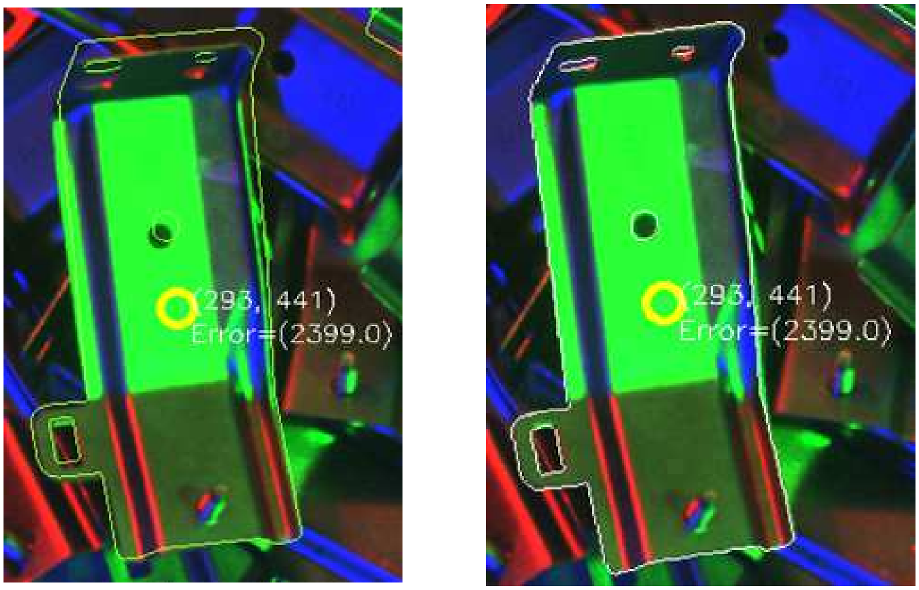


Figure 3.4: Pose hypothesis obtained with the developed algorithm and its refinement. (Left) A pose hypothesis $(\hat{\boldsymbol{x}}, \hat{z}, \hat{\boldsymbol{\theta}})$ is misaligned. (Right) After the pose refinement, the model is aligned accurately. (Best viewed in color)

object pose starting from the discrete pose hypothesis. Assuming that the pose hypothesis is similar to the object pose, an incremental pose update is made by using a visual servoing method. First, we obtain the boundary of the object imaged in a certain pose hypothesis by projecting the object CAD model to the image plane and collecting the 3D coordinates of the boundary of the projected object. Once the computed object boundary is overlaid on the image, we search for the correspondence between that boundary points and edge points extracted from the image. We first compute the direction of the projected boundary pixel and then choose as correspondence the strongest gradient point along its perpendicular, within a small distance. We validate this correspondence by checking the similarity between the boundary direction and the edge direction at the corresponding image point. After establishing all the correspondences between the input image and the CAD model boundary, the 3D object pose is updated by calculating the image Jacobians. This is a conventional visual servoing based object pose refinement procedure [20]. Similar methods are also described in [36]. Further efforts can be put into pose refinement, to avoid getting stuck in local optimal, e.g., as described in [49].

Speeding-up by precalculation. In practice, extracting 3D boundary points by rendering the object CAD model in a given pose takes a considerable long time. To make it faster, we precalculate all the 3D boundary points in each database image in advance. In the refinement process, only the precalculated 3D points are projected. This precalculation makes us avoid the time-consuming boundary point calculation in the iteration loop.

Rejecting hypotheses. Pose estimation using random ferns sometimes proposes wrong pose hypotheses. We use the matching score of the pose refinement as an evidence of the object existence. In searching for the boundary correspondences, we measure the ratio of valid matches out of all the points. If the ratio is less than a certain threshold, we simply reject the pose hypothesis.

4

Experimental results

In Section 4.1 we describe our experimental setup and choice of most relevant parameters. Section 4.2 shows how the developed method works step-by-step, and illustrates pose estimation of different objects. In Section 4.3 we evaluate the quality of our pose hypothesis when varying the number of channels of multi-light images. In Section 4.4 we show how our probabilistic voting approach performs when compared to simple unitary voting. The subsequent sections analyze the system performance in terms of robustness, accuracy, and computation time.

4.1 Experimental setup

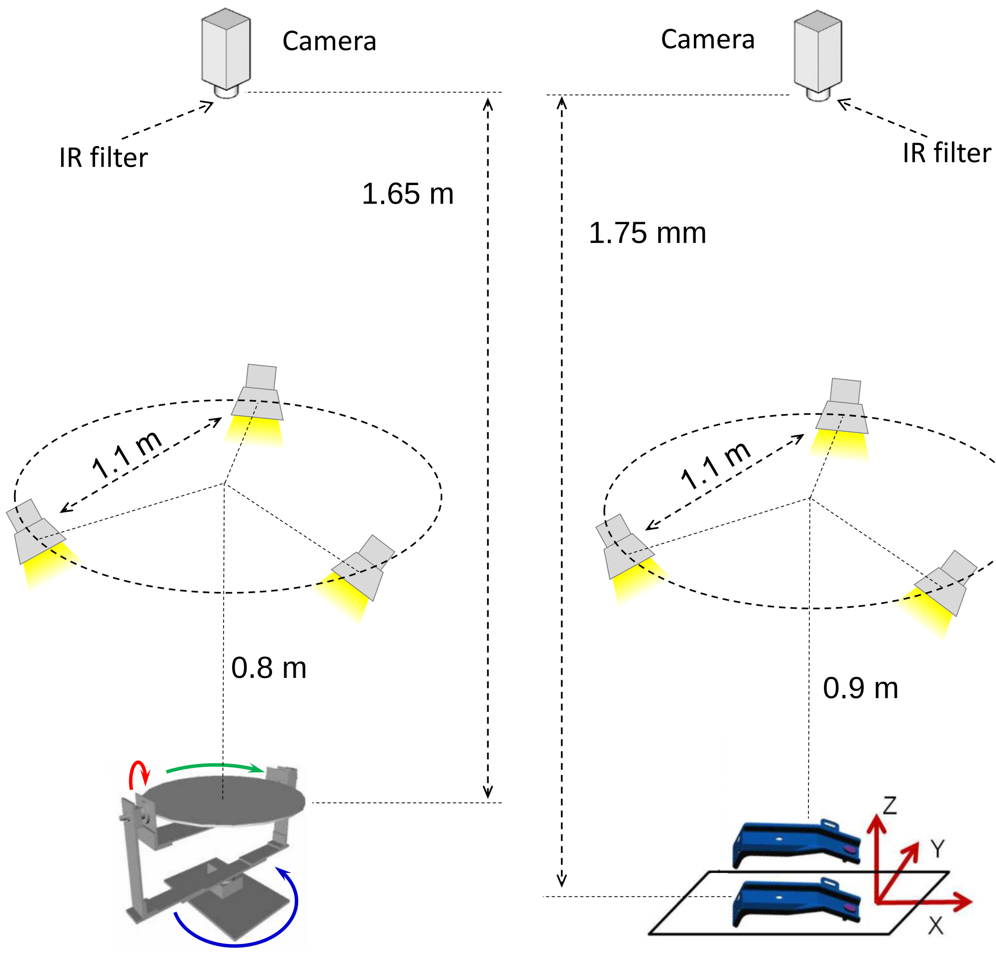
Figure 4.1 depicts the system setup used for training and testing on all our experiments. The system is composed by a B/W camera, and three incandescent light bulbs, which are easily available from a retail store. The lights are about 0.9 meter high from the bottom of the container and roughly located at vertices of a regular triangle. The B/W camera is a PointGrey FL2-08S2M-C, with sensor Sony

ICX 204, of resolution 1024 x 768, equipped with a 16mm lens. The camera is located about 1.75 meters high from the bottom of the container, and aims to the center of the bin. To minimize the effects of the ambient light from fluorescent lights on a ceiling, we acquire infrared images, having a 720nm IR filter attached in front of the lens of the 2D camera. Figure 2.1 shows the real system setup.

Training parameters Using the system in Figure 4.1(a), we generate the database of an object by taking 16602 images, roughly uniformly sampled on $\mathbb{S}^3$, with samples being 4 degrees apart. Then we train one random fern following the training procedure illustrated in Figure 3.3. Since the number of patches to be trained were approximately $1.3x10^8$, we used $m = 27$ fern questions, which generates $2^{27} \approx 1.34x10^8$ fern leaves.

As discussed in Section 3.2, we deliberately over-split the patch space in order to identify non-descriminative training patches or large mismatch between the test patches and the training patches in our database. Empty leaves and leaves with more than 100 patches do not vote for a pose at runtime. Leaves with less than 10 patches vote with 0.1 instead of following Equation (3.14), avoiding aggressive voting as such leaves can represent non-discriminative patches spitted into a separate leave simply due to noise.

Our random fern questions follow Equation (3.13), where positions and channels were created uniformly at random except that questions that are the a repetition of previous ones up to 1 pixel shift in position are ignored and re-created. All our reported experiments used $\tau_i = 10$. To decide on this value we tested two different strategies: using a fixed threshold $\tau = 0$, 5, or 10 for all questions, and randomly selecting τ for each question within the range $[0, 20]$. In almost all the cases there is



(a) Training setup

(b) Test setup

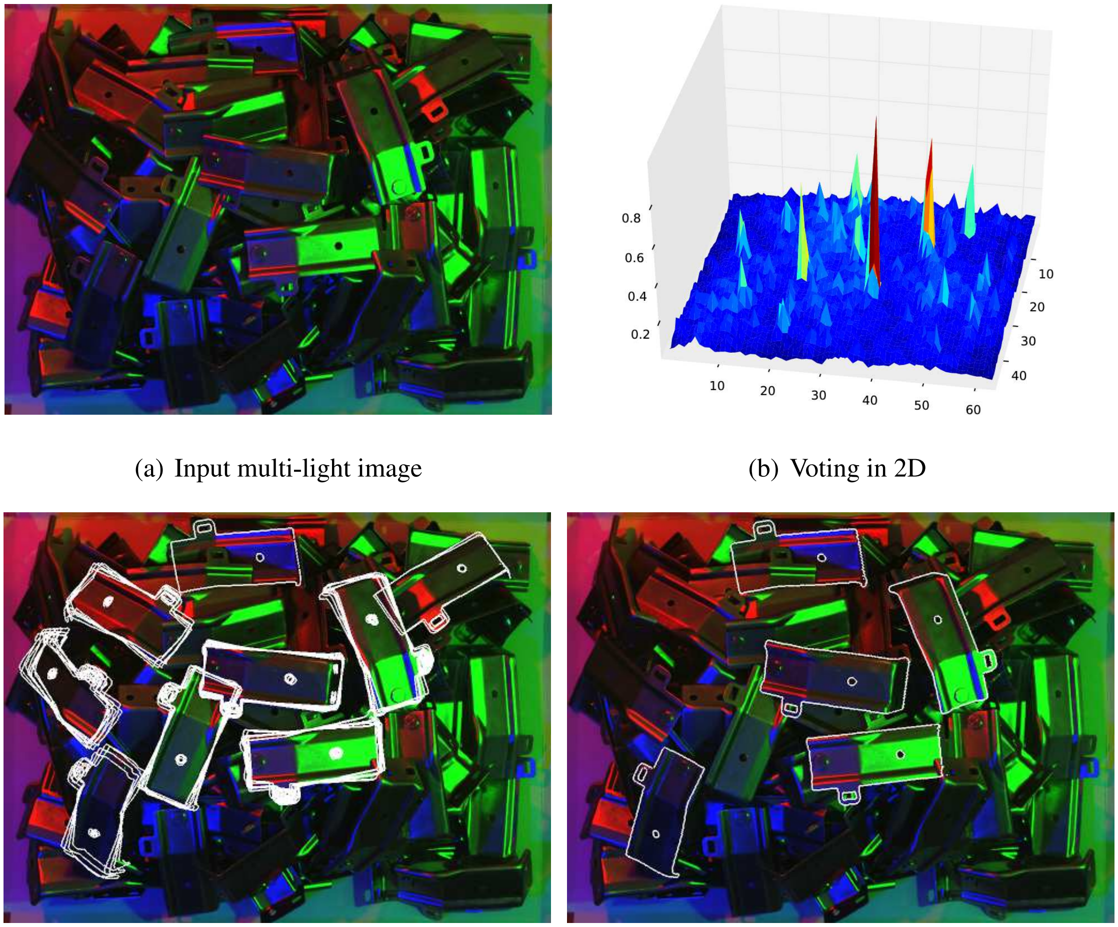
Figure 4.1: Multi-light setup for (a) training and (b) test. Our system works in infra-red wavelength to avoid the influence of environment illumination without the need to cover the system. With our training setup in (a) we create a database of 16602 images, roughly uniformly sampled on $\mathbb{S}^3$ every 4 degrees. Using our test setup in (b) we evaluate the performance of our vision system in the bin-picking scenario. The corresponding real system is shown in Figure 2.1.

no significant performance variation, with the exception of $\tau \approx 0$, where the overall performance degrades a little as the result becomes strongly dependent on image noise when performing questions on uniform regions of the patch with strong gradients.

Pose estimation on sequences of bin-picking images. All our experiments in Sections 4.5, 4.3, and 4.4, use *sequences* of 100 bin-picking images, instead of independently generated random bin-picking images. By using sequences of bin-picking images we keep our experiments as close as possible to the real bin-picking scenario where multiple parts must be picked sequentially from the same bin. This problem is surprisingly much harder than when using independently generated random bin-picking images. After detecting and picking several objects, the subsequent detections and picks are, most often, increasingly difficult because the majority of the remaining objects are the ones that the system kept postponing in the previous cycles as they were harder to tackle.

4.2 Pose estimation examples

Pose estimation process. Figure 4.2 shows the intermediate results of our method, while searching for "bracket" objects. The multi-light image shown in Figure 4.2(a) is captured by the imaging system in Figure 2.1. After voting with each image patch, we observe a few peaks in the 2D voting image $V_I(\boldsymbol{x})$ as shown in Figure 4.2(b). Figure 4.2(c) shows the pose hypotheses selected at the highest peak points, which are fairly accurate. Note that several pose hypotheses can be generated at a single $\hat{\boldsymbol{x}}$ peak. The pose refinement and rejection provide accurate pose estimation in 3D space as shown in Figure 4.2(d).



(a) Input multi-light image

(b) Voting in 2D

(c) 50 pose hypotheses

(d) Top 5 detections

Figure 4.2: Step-by-step pose estimation procedure. We obtained the multi-light image (a) using the imaging system with three lights in Figure 2.1. After voting, the candidate object locations are collected from the marginalized votes (b). Then, pose hypotheses (c) are at the candidate object locations, which are close to the actual object poses. The final detection (d) is obtained through pose refinement.

Support patches. From $V_I(\boldsymbol{x})$, depicted in Figure 4.2(b), we can observe which pose hypotheses have highest voting score. However, it remains unclear the distribution of $(z, \boldsymbol{\theta})$ per peak location $\hat{\boldsymbol{x}}$, and which patches contribute to which peak. To that end, in Figure 4.3(a) we show 3 objects from Figure 4.2(a) with pose hypotheses. For each object, we show in Figure 4.3(b) the 3 pose hypotheses with highest score, and in Figure 4.3(c) which patches contributed to each of those pose hypotheses.

Multi-class object detection. In Figure 4.4, we show the flexibility of our method in a multi-class scenario. Figure 4.4(a) is an input multi-light image which contains two kinds of objects. The search for $N = 10$ peaks results in a few wrong pose hypotheses since the image has only four "bracket" objects, as seen in Figure 4.4(b). Figure 4.4(c) shows that the pose refinement successfully rejects all the wrong hypotheses.

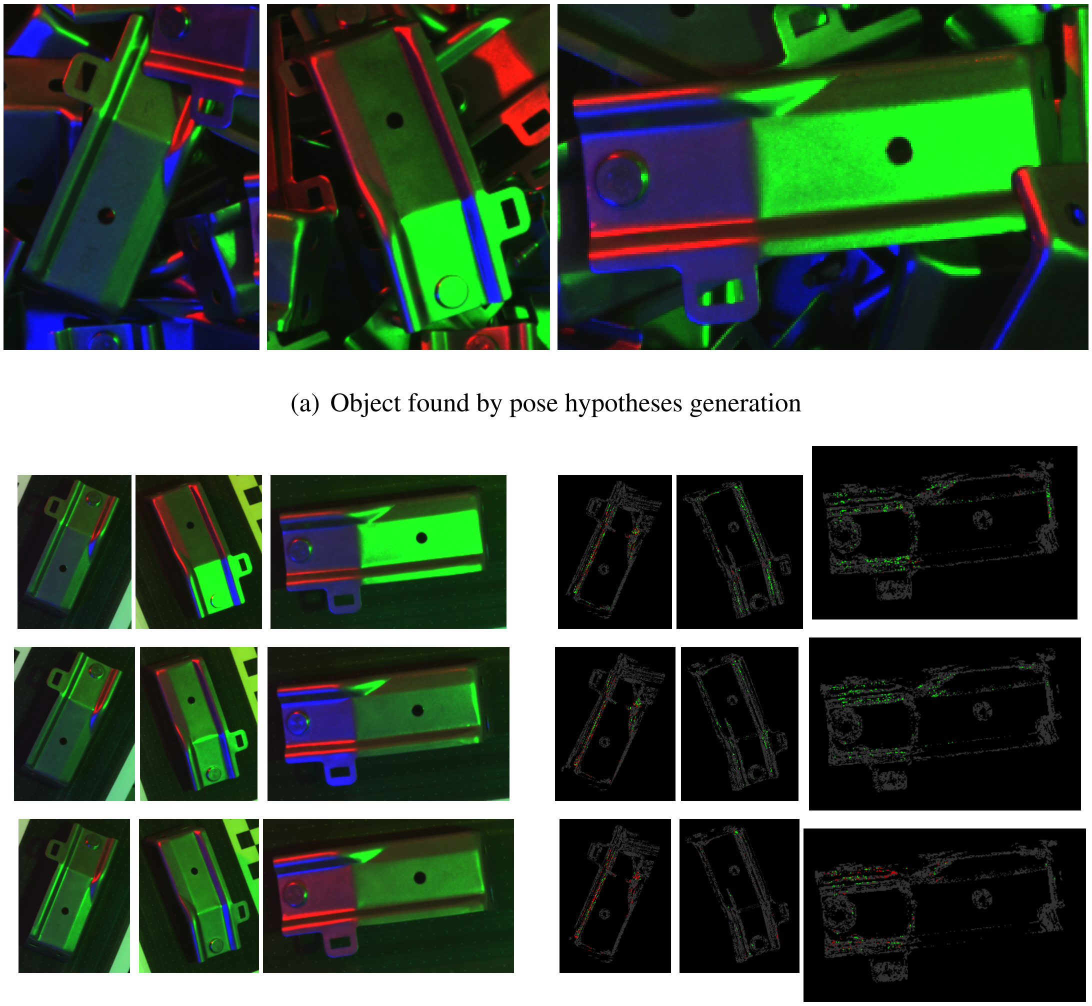
On the other hand, the other objects are successfully detected by just changing the object-specific database, as shown in Figure 4.4(d). This flexibility is very useful for handling many different parts with the same system setup.

4.3 Single-channel images versus multi-light images

To evaluate the benefits of using multi-light images versus single-channel images, we run training and detection on the following 4 types of images:

- *Average image.* A single-channel image created by averaging the channels of our regular multi-light image. It mimics taking a single gray scale image with all 3 lights turned ON.

- *1 channel image.* A single-channel image created by taking the first channel of

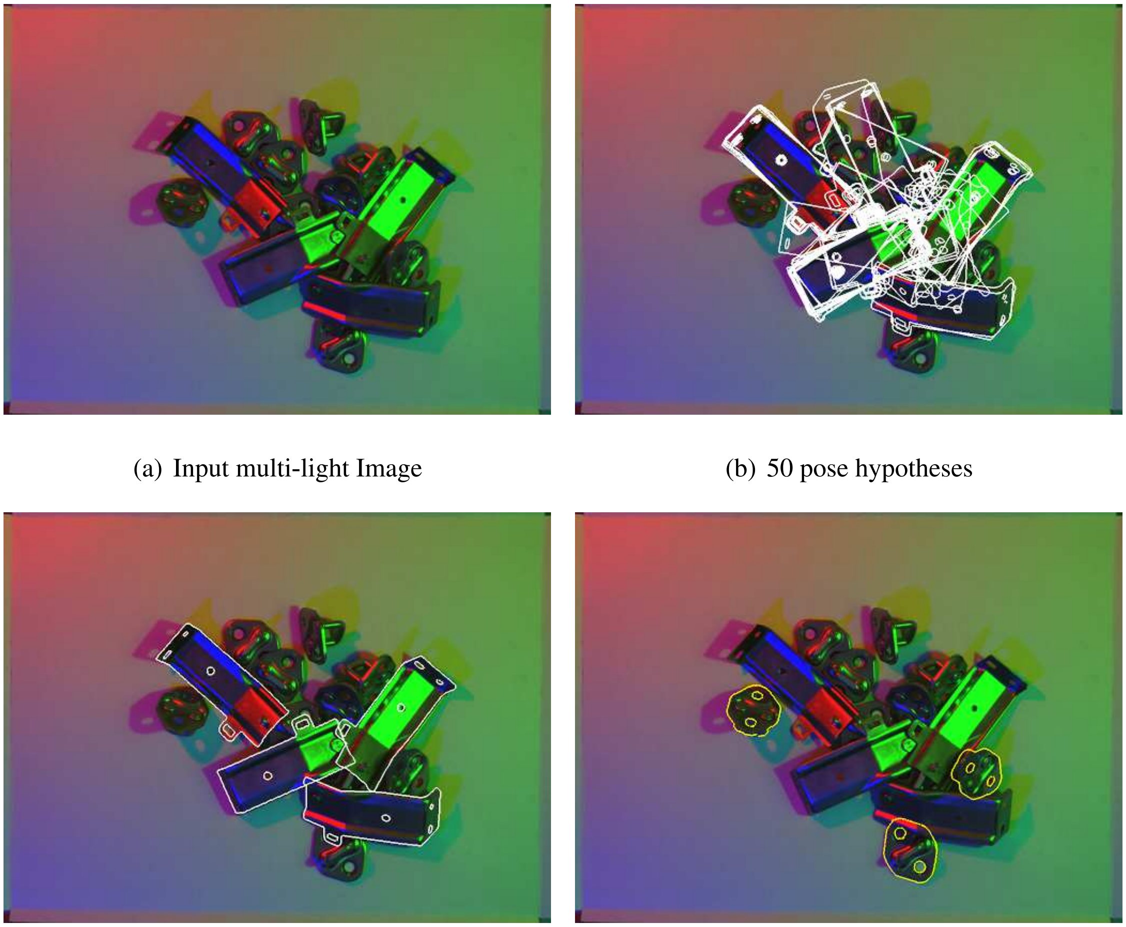


(a) Object found by pose hypotheses generation

(b) Images of 3 pose hypotheses per object found

(c) Patch contribution to each pose hypothesis

Figure 4.3: Illustration of pose hypotheses and patch contributions for some of the objects in Figure 4.2(a). (a) 3 objects from Figure 4.2(a), (b) best 3 pose hypotheses found, and (c) patches contributing to each of the pose hypothesis found. In (c), green pixels correspond to locations containing votes from the pose hypothesis found, red pixels correspond to locations containing votes from supporting hypothesis pose neighbors, and gray pixels correspond to voting locations with no votes from the pose hypothesis found or its neighbor poses.



(a) Input multi-light Image

(b) 50 pose hypotheses

(c) Detection results

(d) Detection of another object

Figure 4.4: Object specific detection. (a) The scene has two kinds of objects, and (b) some of the pose hypotheses are incorrect. (c) The pose refinement successfully rejects the incorrect hypotheses. (d) The exactly same code can be used for another object by simply changing the object-specific database.

our regular multi-light image.

- *2 channel image.* A dual-channel image created by taking the first 2 channels of our regular multi-light image.

- *3 channel image.* Our regular multi-light image with all 3 channels.

We use the same training dataset and 500 detection images as in Section 4.5, since all the types of images listed above can be generated from our multi-light images. Our training and detection procedure remain the same.

To evaluate the difference in detection capability between using single-channel images and multi-light images we define the ratio between the highest peak of $V(\boldsymbol{x}, z, \boldsymbol{\theta})$ and the average of all the voted (non-zero) $V(\boldsymbol{x}, z, \boldsymbol{\theta})$ entries as peak-to-noise ratio (PNR).

Table 4.1: Peak-to-noise ratio of probabilistic voting with single channel and multi-light images

Rank	average image	1 channel image	2 channel image	3 channel image
1	29.17	29.02	49.84	**72.31**
2	24.40	23.89	38.20	**50.24**
3	22.11	21.32	33.26	**42.45**
4	20.74	19.90	30.57	**38.23**
5	19.66	18.71	28.51	**35.59**

For all 500 bin-picking picking scenes, we generate the 4 types of images listed above and we plot in Figure 4.5 the PNR of probabilistic voting on those images.

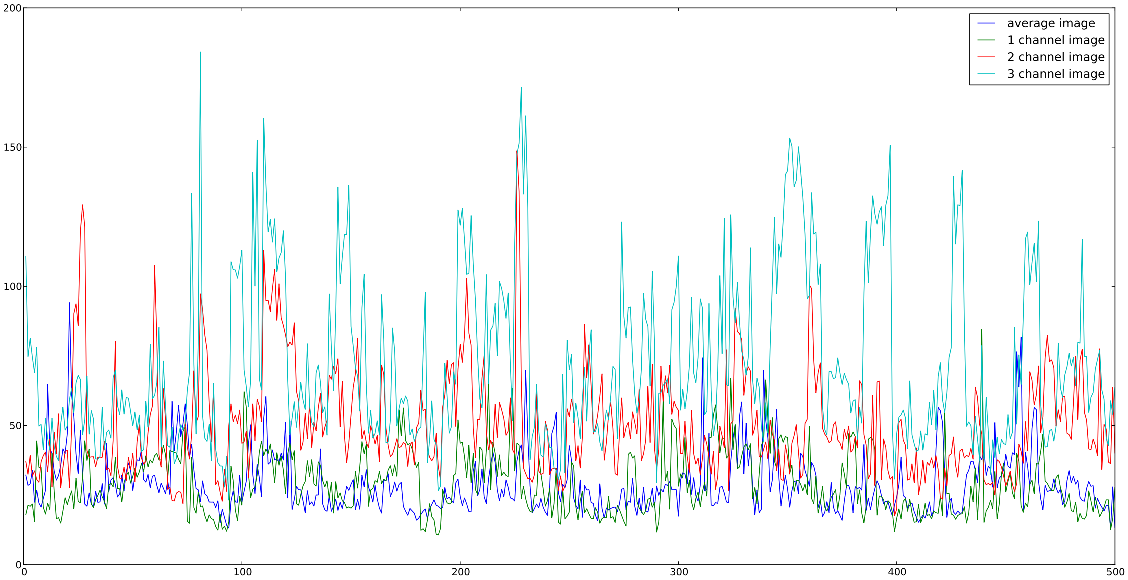


Figure 4.5: Peak-to-noise ratio of probabilistic voting with single-channel and multi-light images. While single-channel images with one or all three lights ON produce poor voting peaks, multi-light images generate much stronger voting peaks. Multi-light images creates a much larger number of discriminative patches, notably boosting the voting procedure.

We also summarize, in Table 4.1, the average PNR across all 500 picking scenes, per image type and for the 5 highest voting peaks.

The difference in the results lie on the richness of the content of each type of images. Since each channel of a patch contains large homogeneous regions, it is hard to generate a large number of discriminative labeling questions from a single channel. On the contrary, by using multi-light images, with channels varying differently with surface orientation, we increase the discriminative power of the labeling questions obeying Equation (3.13) by providing additional pose information on the new channels. Consequently, we increase the quality of the probabilistic votes in each leaf of the fern, during the training procedure illustrated in Figure 3.3. Figure 4.5 and Table 4.1 show the clear benefits of using multi-light images in the voting procedure as it shows much higher scores for multi-light images than for single channel images.

4.4 Probabilistic voting versus unitary voting

In order to illustrate the capabilities of our probabilistic voting approach, we repeated the experiments made in Section 4.3 with a change in the voting procedure. Now, instead of following the probabilistic votes of Equation (3.14), we perform unitary voting, i.e., all the patches vote for poses with the same weight 1.

We display the PNR of unitary voting in Figure 4.6, with the same scale as Figure 4.5 for easy comparison. We also summarized the PNR of unitary voting of the 5 highest peaks of $V(\boldsymbol{x}, z, \boldsymbol{\theta})$ in Table 4.2.

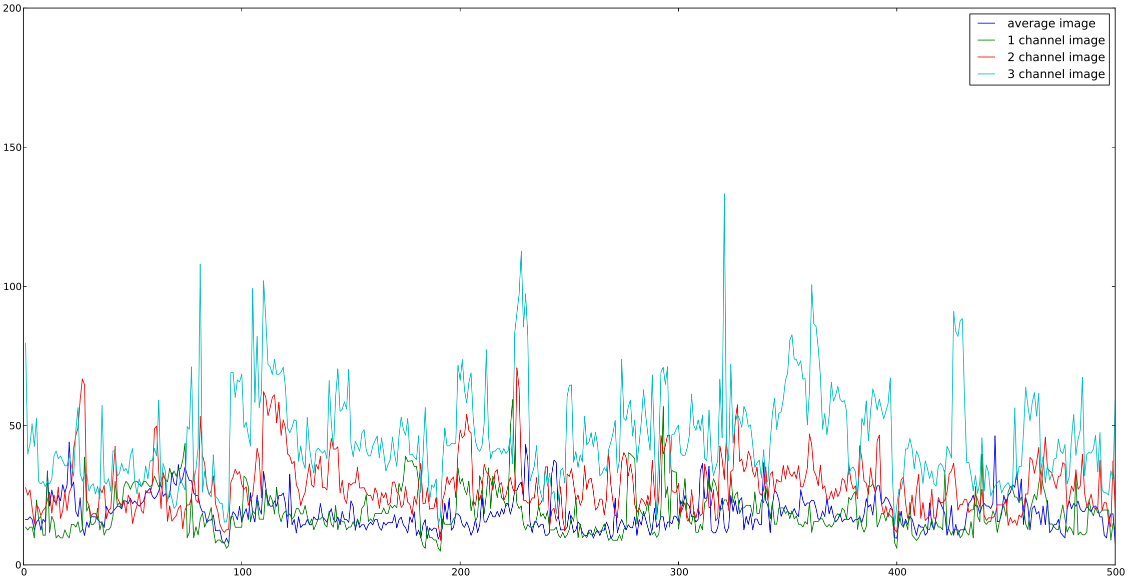


Figure 4.6: Peak-to-noise ratio of non-probabilistic voting with single-channel and multi-light images. In comparison with the PNR values of probabilistic voting in Figure 4.5, the PNR values of non-probabilistic voting are considerably lower.

Table 4.2: Peak-to-noise ratio of unitary voting with single channel and multi-light images

Rank	average image	1 channel image	2 channel image	3 channel image
1	18.56	18.91	28.09	**45.38**
2	16.03	16.90	22.43	**33.97**
3	14.91	15.10	20.03	**29.63**
4	14.12	14.15	18.64	**27.24**
5	13.47	13.40	17.65	**25.59**

By comparing the PNR values with the ones of the previous section we can see the benefits of using probabilistic voting versus unitary voting. The PNR values in Table 4.1 are consistently higher —1.53 times higher on average —than in Table 4.2. These results show that it is not only important to generate images with rich pose cues but also to make sure that the patches of the input image that are more distinctive vote with more certainty.

4.5 Detection performance

Table 4.3: Detection performance of 100-part picking test

Rank	Number of detections	False alarms	Inaccurate poses
1	498 (99.6%)	1 (0.2%)	11 (2.2%)
2	447	4 (0.89%)	20 (4.47%)
3	341	1 (0.29%)	13 (3.81%)
4	218	4 (1.83%)	11 (5.04%)
5	96	1 (1.04%)	6 (6.25%)
Total	1600	11 (0.7%)	61 (3.8%)

Our multi-light approach. To test the detection performance of our method, we design a “100-part picking” test. At first, we randomly stack 100 parts in a bin, within the image field of view. Then, we run our method to detect object poses, and we carefully pick out, by hand, the object with the highest detection score. We repeat the detection procedure on the remaining parts in the bin, and continue to pick out parts until we empty the bin. We conduct this test five times, processing 500 images in total. Our method fails to detect object poses only in two images out of 500, and only one pose with the highest detection score is a false alarm. While trying to detect 5 poses per image, a total of 1600 poses are detected, with an overall false alarm rate of 0.7%, and the remaining poses are rejected after refinement. Within this experiment, the pose estimation never fails for images with less than 5 objects. Sometimes, pose refinement is trapped by the nearby strong image gradient. It happened 3.8% in total and 2.2% for the best pose detected. Table 4.3 shows the statistics of the 5 best pose hypotheses in each of our 500 bin-picking images.

Hierarchical template matching. We compare the detection performance of our method with the commercial implementation HALCON developed by MVTec. To this end, we use the 5 sequences of 100 images previously collected, and we detect one part per image. Since a sequence of 100 images is generated by sequentially picking each object detected by our system, a different detection by HALCON implementation often results in multiple detections of the same object until it is removed. Thus, to measure the detection performance, we just account for the 182 images containing the first detection of an object, removing all the repeated detections. Such experiment gives to the HALCON implementation accurate detections with the rate of 92.31% on the 182 images being considered. This result is considerably lower than the results obtained with our system, where we have accurate detections with the rate of 97.2% for 500 images, reported in Table 4.3, or $98,35\%$ for the set of 182 images being used in this HALCON experiment.

Figure 4.7 shows a few incorrect results obtained with the HALCON implementation, illustrating that the exhaustive search for location and local direction of shape edges is not enough to obtain good results in a bin-picking application. Template matching often gets a high matching score while matching edges from multiple objects, substantially decreasing the performance in presence of partial occlusions, even when fully visible parts exist in the image. Our approach makes use of the photometric appearance of the object under different poses to overcome such limitations present in edge-based template matching approaches.

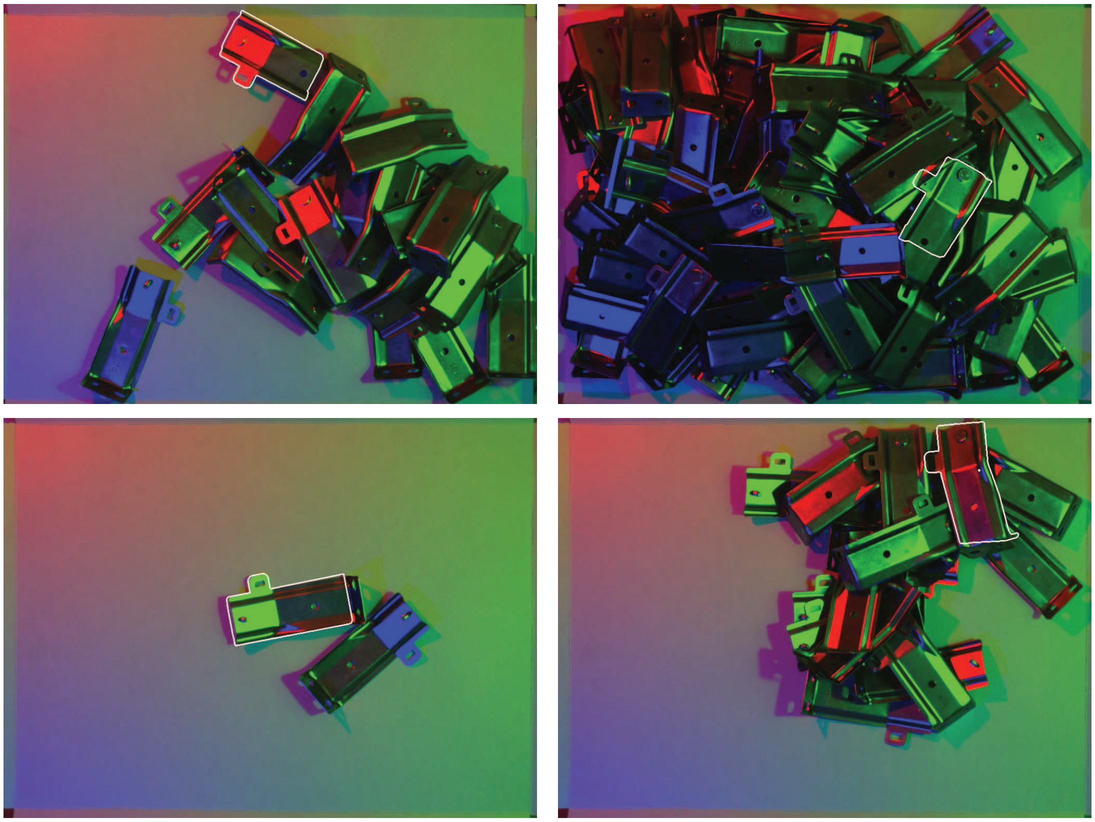


Figure 4.7: A few wrong detections obtained with the template matching approach, illustrating that the exhaustive search for location and local direction of shape edges is not enough to provide accurate detections and eliminate false positives, even in cases of no occlusion, no shadow, and no near inter-reflection.

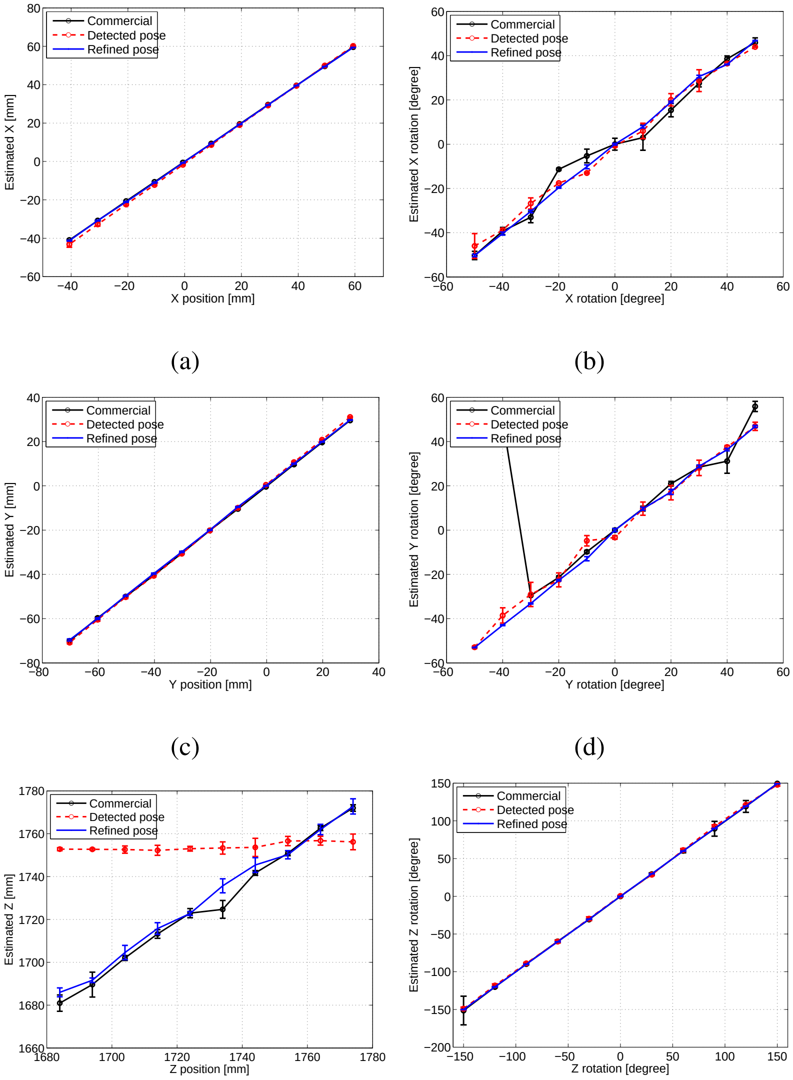
4.6 Pose estimation accuracy

As we discussed in Chapter 2, the light sources in the system are not far enough to provide parallel illumination, due to which surfaces in the same orientation at a different location may have different colors. Because of such location dependency and non-orthographic projection, the database image usually differs from the input object image, even though the object orientation is the same.

To test the accuracy of our pose estimation with respect to object location, we conducted an experiment where we vary the orientation and location of the object. We located an object right under the camera at first, and moved it using a rotation stage and a linear guide along each axis. For each location and orientation set, we compare the estimated pose with the pose obtained from the ground-truth of the rotation stage and linear guide. To study the repeatability of the result we collect 100 multi-light images. Figure 4.8 shows the statistics of the results obtained with our method, before and after pose refinement, and with the HALCON implementation.

Because the camera is located at 1750 mm high, which is much larger than the object size, the estimated rotation around X and Y axes and the translation along Z axis are less accurate than the rotation around Z axis and the translation along X and Y axes, for both our method and the HALCON implementation. We notice that the translation along X and Y axes and the rotation around Z axis are very accurate even though the multi-light images are location-dependent. This is because our method does not use the absolute intensity of pixels, but only rely on the intensity difference in each channel, as discussed in “Dealing with multiple channels” of Section 3.2. In addition, placing the camera far from the scene decreases the perspective distortion and visual servoing successfully corrects the coarsely computed pose hypotheses.

The estimated pose obtained from the voting process is one of the discretized pose hypothesis from the object database. Pose refinement improves the accuracy of the estimated pose significantly by correcting the pose errors due to discretization. Even though rotation around X and Y axes and the translation along Z axis are visibly inaccurate after the voting process, as seen in Figure 4.8, pose refinement greatly increases their accuracy. We observe that the large errors in the estimated Z location do not significantly affect the accuracy of pose refinement. This is because such large errors in Z are perceived as minor changes in the size of the imaged object, due to the large distance of the bin to the camera.



(a) (b)

(c) (d)

(e) (f)

Figure 4.8: Analysis of pose estimation accuracy. The camera is placed at about 1750mm high from the object and the axes are set as shown in (a). We show the pose estimation results of our method, before and after refinement, while changing the object location (b,c,d) or orientation (e,f,g). We also show the results of the HALCON implementation. For each pose, we acquired 100 images and computed the mean and variance of the obtained pose.

4.7 Speed

Table 4.4: Processing time for detecting up to 5 poses with 50 pose hypotheses

Estimation phase	Average [ms]	St. dev. [ms]
Choosing voting points	69.1	6.7
Labeling and Voting	136	41.7
Generating pose hypotheses	75.4	21.6
Refining pose hypotheses	164	35.6
Total	445	76.5
HALCON [-30°,30°]	839	101
HALCON [-50°,50°]	2446	243

Table 4.4 shows the computation time in each estimation phase for detecting up to 5 poses. We used a 3.2GHz Intel QuadCore processor with 3GB memory for this test, without implementation of CPU intrinsics or usage of additional computational hardware. In this case, at most 50 pose hypotheses were tested after picking up 10 peaks in the marginalized 2D voting image $V_I(\boldsymbol{x})$. We used a set of 100 bin-picking images for this analysis. Speed of the labeling process depends on the number of voting points, which are determined by the complexity of the input image. In average, the whole process is done in about 500 ms per image. Refining and evaluating poses takes the longest time, being linearly proportional to the number of pose hypotheses. If a user wants to detect just one pose, the number of hypotheses can be reduced. Compared to the commercial implementation HALCON, our method runs faster.

The processing time of the HALCON software using template matching depends on the pose coverage. For similar pose coverage of [-50°,50°], our method runs more than 5 times faster.

5

Conclusion

As stated in the introduction, the main goal of this work is to design a practical 3D pose estimation system for the bin-picking of shiny and textureless objects. We develop a multi-light acquisition system and a fully data-driven method to jointly tackle the bin-picking problem. The accomplishments of our work are summarized as follows:

- ***Creation of images with rich pose clues using a multi-light imaging system.*** We use a multi-light imaging system, as in photometric stereo, to create object images with rich pose clues. Using such multi-light system, we are able to explore the photometric properties of a textureless object to estimate its 3D pose. Our system is cheap, easy to setup, and it works under unrestricted lighting conditions and unknown illumination directions.

- ***Probabilistic voting for pose estimation.*** We design a new probabilistic voting framework to map the appearance of image patches into votes for the object pose. The voting method is robust to changes in the object appearance due to object translations, where the incident and viewing angles can vary substantially. Our method is also robust to appearance variations due to shadows, inter-reflections, and occlusions. The method handles distinct objects by simply changing the image database.

- *Accurate pose estimation.* We refine the 3D pose hypotheses obtained in the voting step by using a model-based tracking technique, common in 3D visual servoing [20], achieving accurate object poses for robotic manipulation.

- *Benchmarking.* We compare our system with the HALCON software, a highly optimized commercial solution based on hierarchical template matching. Our system is more than five times faster and provides superior detection results with just a single fern.

While focusing our discussion on the detection and pose estimation of textureless and shiny objects, our approach remains generic. Thus, we strongly believe that our approach can be applied to a large class of objects including the ones that are textured in certain viewpoints and textureless in other viewpoints, without the need for a multi-modal approach.

6

Discussion

A single bin-picking system working in an assembly line may need to perform hundreds or even thousands of picks per hour. As a result, even though our system computes an accurate pose for 97% of the bin images, we may still require a picking system with strong recovery mechanisms or frequent intervention of a human operator to cover the limitations of the bin-picking system. In order to tackle the major issues that adversely impact the accuracy and usability of our system, we discuss possible extensions to our current approach. In addition, the system requires acquiring a large set of multi-light images before beginning to recognize a different object. Hence, we identify potential ways of shortening the acquisition time, to increase the usability of our system.

6.1 Topics for future research

Alternative patch representations. Random ferns are a computationally inexpensive way of creating patch descriptors and providing efficient nearest neighbor search, allowing us to compute them in a dense number of points over the input image. However, we can consider other patch descriptors such as HOG [48] (Histograms of Oriented Gradients), D-Brief [47], or CNN-based descriptors [46]. Although more expensive to compute, compare, and store, they are alternative patch representations that have shown to be robust to several image disturbances such as noise, illumination changes, and small image transformations. Therefore, they have the potential to improve the mapping from patch appearance to pose space, and enabling the learning of more meaningful probabilistic votes for pose hypotheses generation. With alternative patch representations, we can also consider other similar approximated nearest neighbors search structures, popular with such kind of patch representations, based on hierarchical k-means trees, k-dimensional trees, hash maps, etc.

Ensemble learning. Using forests instead of a single decision tree has shown to improve object recognition and pose estimation results [6]. The number of decision trees to use in a forest is a tradeoff between accuracy, memory usage, and run-time. We expect the labeling and voting time of $136ms$, reported in Table 4.4, to increase linearly with the number of ferns used, while keeping the remaining processing times nearly constant.

Multiplexed illumination for noise reduction. In our system, we acquire images under variable illumination directions using only a single light source at a time. Another way to obtain the same images is based on a multiplexing principle [39, 40]. In such a method, images are acquired with multiple light sources simultaneously illuminating the scene from different directions, which can be computationally demultiplexed using the same number of images as in the single light source method. For N light sources, the multiplexed illumination method increases the signal-to-noise ratio by $\sqrt{N}/2$, allowing faster acquisition times or less noisy inputs for the computer vision algorithms. This approach is useful for imaging dim object areas, common in shiny objects when illuminated by small light sources.

Video-rate acquisition of multi-light images. Capturing multiple channels of the multi-light image can be done in parallel by frequency division multiplexing instead of time division multiplexing. For example, using a 3CCD camera (or any other multi-spectral camera with non-overlapping bands) and 3 colored lights at the frequencies of maximum sensor response for each channel band, we can acquire a multi-light image with a single snapshot. This way, we can acquire a large set of training images at video rate and have a very fast acquisition system for real-time pose estimation and picking. Also, we can avoid the otherwise unpleasant flashes coming from a multi-light imaging system.

Illumination and acquisition strategies. Illuminating the scene with three light bulbs from distinct directions, as in photometric stereo for lambertian surfaces, is only one of the many possible ways of illuminating the scene to extract 3D information or reflectance information. Surely the fields of shape from shading, multiplexed illumination, and relighting, have shown many other ways of exploring different illumination patterns and a diverse number of lights that should be explored for bin-picking. Light-field photography is also a known way to explore light to more easily collect 3D scene information or reflectance distribution functions of the scene surface.

Bin-picking for large bins. When picking in large volumes, the learned object appearance can differ drastically from the appearance in the input image for the given illumination. To tackle such issues, one can design approximately parallel illumination that would provide constant illumination direction and power across the bin. However, such illumination designs are usually not compact and not so affordable. Instead, we can normalize the image per average illumination power per image ray, as well as derive expected approximate changes in appearance per pixel (or per pixel region) and per channel instead of assuming them constant across all the large bin volume.

Multi-light bin-picking should meet CNN. With the rising of deep Convolutional Neural Networks (CNN), the advances in object classification, detection, and instance segmentation have been remarkable. However, the problem of 3D pose estimation from monocular images have received limited attention from the deep learning community until very recently [53, 54, 55, 56, 57, 58, 59]. While the experimental results look promising, they tend to fall short on real bin-picking data like the one presented in this work. It would be great to see results on object-specific bin-picking where the clutter in the scenes have the same properties as the object to detect. Also, experiments with deep bins and textureless shiny objects would make more evident the power and drawbacks of approaches regarding strong occlusions, shadows, inter-reflections, instability of appearance, and scarcity of distinctive features. Last, it would be great to see how such approaches could benefit from multi-light images.

Multi-view pose estimation. Capturing multiple views of an object can be useful to speed up the acquisition of the object database and improve the robustness of the online algorithm. In order to accelerate the acquisition of the object database, we can place the object in a smaller number of poses while acquiring more views of the object per pose. During the online process, we can use the voting procedure on the multiple views of the bin-picking scene, possibly improving the object detection. Also, pose refinement and evaluation can be made more robust since the object pose can be more accurately computed, e.g., due to better depth perception and increased geometrical constraints, possibly reducing the number of inaccurate poses and false positives reported in Table 4.3.

Benchmark on very large sets of objects. One of the very limiting factors of deploying scalable picking systems to factory automation and logistics is the difficulty in creating generic systems that can recognize and manipulate a multitude of objects. While we focus on the very hard problem of bin-picking of textureless and shiny objects, we believe that our approach remains generic for a much broader class of objects containing textured and/or textureless viewpoints. Thus, we see great value in evaluating our approach on very large datasets and tackling issues related to the large variations in object properties.

Bin-picking with real-time 3D data. Major efforts are still needed to explain every single pixel of the unstructured scene inside a bin from 2D vision, in order to meet the very-low-failure requirements of industrial bin-picking applications. Simply estimating poses using a monocular 2D camera is not enough for safe robot manipulation as the rest of the bin-picking scene remains unexplained and sporadic incorrect poses can make the picking system unreliable, hence unusable. We concede that 3D data is very useful to further prune votes, early reject pose hypothesis, aid pose evaluation and refinement, as well as provide dynamic obstacle avoidance for robot motion planning. However, even though current industrial 3D vision sensors for bin-picking provide fairly accurate pointcloud, they remain very expensive and slow. They are mostly based on structured light, slow due to the need of acquiring long sequences of patterns, or on stereo-matching, fast in acquisition but computationally very intensive, using time and system resources that could be otherwise used for pose estimation. Instead, we believe that, with re-doubled effort in monocular pose estimation, cheap and real-time 3D sensing, e.g., time-of-flight (ToF) sensors, can be sufficient to complement (rather than almost substitute) 2D vision while remaining affordable, thus scalable, and real-time. This idea contradicts the current trend in vision systems for 3D pose estimation [9, 41, 42, 43, 44, 45, 51] that heavily rely on 3D data.

Structured learning. Rather than having patch labels computed independently of each other, we should obtain coherently labeled regions [52]. Such approach should be able to correct erroneous noise-prone label bits to some extent, substantially guiding voting towards an improved peak-to-noise ratio.

Factoring out inter-reflections for appearance stability. The appearance of an object in a given pose can vary substantially on changing the surrounding scene, in large part due to shadows and intra and inter-reflections, making pose estimation unreliable. Thus, we highlight the usefulness of computing the direct component of illumination [21, 38], factoring out intra and inter-reflections and, consequently, easing shadow detection. By computing the direct component of illumination, we can make our system more predictable and trustworthy at localization and evaluation stages, since each pixel intensity becomes more closely related to the orientation of the object surface and reflectance model. Thus, the realistic rendering of new object views will become easier, enabling acquisition of fewer training images and rendering of new views for more precise pose hypothesis and evaluation. Given the clear advantages of removing inter-reflections, we detail below this possible path of future

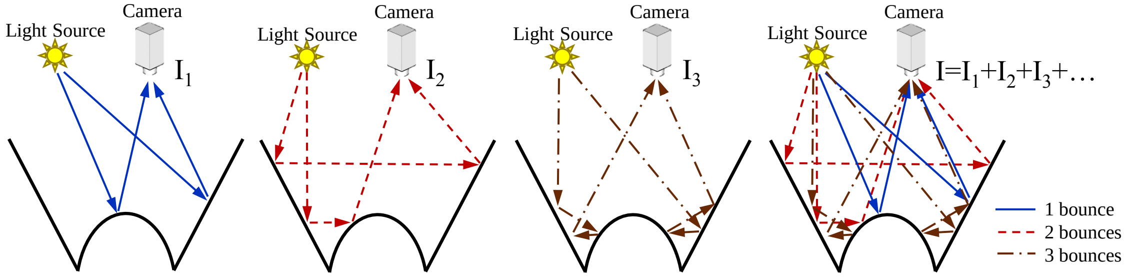


Figure 6.1: Illustration of light transport using a single light source. The intensity of an image pixel results from a complex sequence of reflections and inter-reflections, since the light emitted from a light source may bounce on one or more surfaces before it reaches the camera. Only the intensity of the first bounce, i.e., the direct component of illumination, can be independent of the remaining scene.

research. While we believe that the discussion below opens doors to more precise, physics-based, computer vision, we stress that its applicability remains limited due to the current pricy and complex approach in direct-global light separation.

6.2 Direct-global light separation: tackling inter-reflections

As illustrated in Figure 6.1, the appearance of a surface point is usually influenced by reflections from the surrounding scene. In a bin-picking scenario, the directional shape of the BRDF of shiny objects and the proximity among objects makes such appearance changes even more visible. Also, the possible environment changes, e.g., people or robots moving nearby or different system placement with respect to walls, can considerably influence the object appearance.

Therefor, we discuss below how to compute the direct component of illumination of our bin-picking images, factoring out appearance uncertainties due to inter-reflections. To obtain the direct component of illumination, we lit the objects in the bin with high-frequency spatial patterns, using projectors instead of simple light bulbs. The direct component of illumination provides each pixel value with a more straightforward physical interpretation, since their intensities will be more closely related to the BRDF and normals of the object surface. These pixel intensities would allow us to easily detect shadows, redesign tree questions based on pixel color and presence of shadows, realistically render new object views, among others.

Related work. The most popular methods that estimate the direct component of illumination aim to compute the transport matrix [35], after which the inter-reflection components can be computed and removed. However, the acquisition and computational effort needed to compute such matrix would make our system impractical. Since we just need to separate the direct illumination component from all the other components, we can use the method in [21, 38] to factor out inter-reflections and other global effects from each of our light sources. We describe a simplified version of the original method in the sequel, and discuss the advantages and limitations in using it in a bin-picking system.

Computing the direct component of a scene via high frequency illumination. Consider a scene viewed by a camera and illuminated by a point light source. The radiance R of a point in a scene, measured by the camera, can be decomposed in direct and global components as

$$R = R_d + R_g, \tag{6.1}$$

where R_d and R_g are the direct and global component of the radiance R.

Now, assume that only half of the light source pixels are activated and that these activated pixels are well-distributed over the entire scene to produce a high frequency illumination pattern. Under the smoothness conditions discussed in [21], where the light transport coefficients remain approximately constant within a neighborhood, we can write that

$$R^{\text{on}} = R_d + 0.5R_g \quad R^{\text{off}} = 0.5R_g, \tag{6.2}$$

where R^{on} and R^{off} are respectively the radiances of a lit or unlit point in a half-illuminated scene, and R_d and R_g are, as in Equation(6.1), the direct and global components of a point when the scene is fully illuminated. Under the mentioned high-frequency spatial illumination and scene smoothness assumptions, if only half of the scene is illuminated, the contribution of the scene to the global component of the point radiance under analysis is also half. Also, $R^{\text{on}} \geq R^{\text{off}}$, since R_d is non-negative.

In order to compute the direct component of illumination, consider two checkerboard illumination patterns with very tiny squares (high-frequency), that have only half of the source pixels activated and that lit with complementary illumination, i.e., if both the checkerboards simultaneously illuminate the scene, it will be equivalent to having the light source fully turned on. With these two illumination patterns, it is possible to obtain the values R^{on} and R^{off} for each pixel in the scene, since the pixels that are lit with one illumination pattern are unlit with the complementary pattern. For the two values R^{on} and R^{off} that we obtain for each pixel, we know that $R^{\text{on}} \geq R^{\text{off}}$, thus the larger value corresponds to the case where the pixel is lit, and we can compute the direct component of illumination as

$$R_d = R^{\text{on}} - R^{\text{off}} = R^{\max} - R^{\min}. \tag{6.3}$$

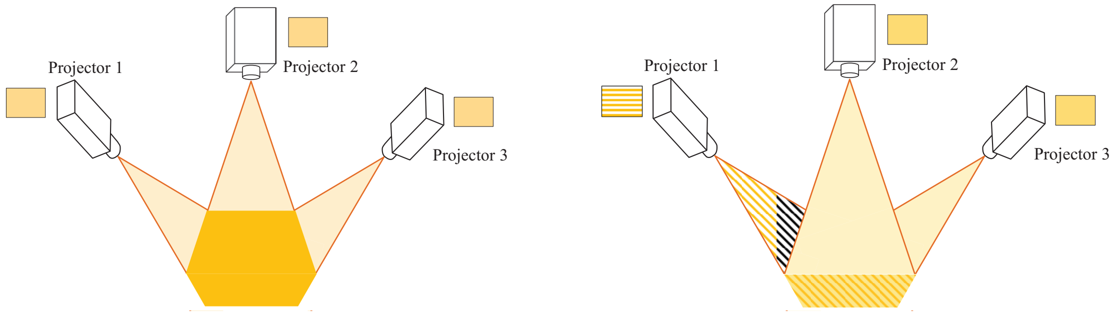
One main disadvantage of illuminating shiny surfaces with point light sources, e.g., a projector, is that it accentuates the specularities and micro-texture of the surface material. To reduce such effects, we can enlarge the light source, while still being able to project high frequency patterns, by combining a projector with a linear diffuser positioned perpendicular to one-dimensional patterns, as in [37].

Computing the direct component of illumination in a multi-light setting. The method described above requires 2 images per light source in order to compute the direct component of illumination. In a multi-light setting, this corresponds to taking $2N$ images when using N light sources. A theoretical result in [38] shows that, instead of $2N$, $N+1$ images are sufficient. Let R_d^i and R_g^i denote the direct and global illumination components corresponding to the i-th light source $i = 1, \cdots, N$. First, we turn on all N lights at half-brightness and compute the image I_0:

$$I_0 = \sum_{i=1}^{N} (R_d^i + R_g^i)/2. \tag{6.4}$$

Next, we set the i-th light source to be a high frequency pattern, with half the pixels on, while keeping all the other $N-1$ light sources at half brightness. Since the average intensity of the high frequency patterns is half of their maximum intensity, the global component of a light projecting such high frequency pattern is the same as that of a light homogeneously illuminating the scene at half-brightness, i.e., $R_g^i/2$. Thus, in the captured image intensity L_i, only the coefficient of direct component changes, obtaining

$$L_i = \begin{cases} L_0 + R_d^i/2, & \text{for lit points} \\ L_0 - R_d^i/2, & \text{for unlit points.} \end{cases} \tag{6.5}$$



(a) Illumination for L_0

(b) Illumination for L_1

Figure 6.2: Illustration of how to acquire images in a multi-light setting in order to compute the direct component of light. Example for $N = 3$ light sources. The projectors simultaneously illuminate a scene at half brightness to create L_0. Then, to create L_i, the i-th light source is set to be a high frequency pattern, with half the pixels on, while keeping all the other $N - 1$ light sources at half brightness.

Since $I_i - I_0 = \pm R_d^i/2$, the N direct components can be computed as

$$R_d^i = 2\,|L_i - L_0|\,, \quad i = 1, \cdots, N, \tag{6.6}$$

from $N + 1$ images.

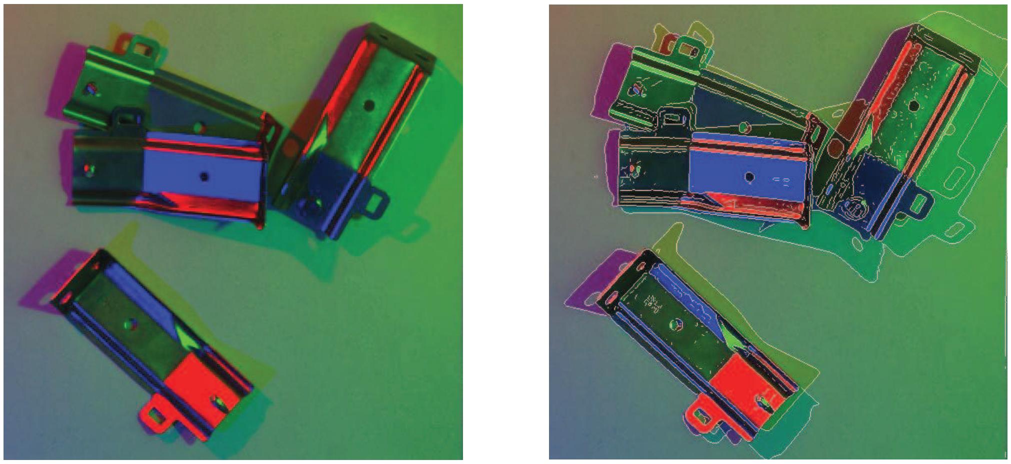
Benefits of computing the direct component of illumination. By computing the direct component of illumination, our data-driven approach can take advantage of several facts:

- *Shadows become easier to detect.* Our voting procedure should take into account the existence of shadows when searching for similar patches, as illustrated in Figure 6.3. In order to detect shadowed areas, it is common to assume that they display less radiance than areas being directly illuminated. However, inter-reflections can cause shadows to display more radiance than directly illuminated surfaces, making it hard to discriminate between shadows and directly illuminated surfaces. Computing the direct component of illumination removes inter-reflections, due to which shadows become easier to detect since they will display no radiance.

- *Stability of appearance.* By using the direct light component, the appearance of the object surfaces no longer depends on the surrounding scene. Thus, neither the surrounding bin objects nor the environment changes such as bin walls or objects outside the working area will affect the surface appearance. As a result, the use of the direct light component makes the object appearance in training images and test images easier to compare, making our system more predictable and trustworthy at localization and evaluation stages.

- *Advantages of acquiring the BRDF of the object surface.* The direct light component of an illuminated patch directly provides BRDF information of a surface. Obtaining the BRDF of an object surface opens the doors for research in the acquisition of less training images, rendering of new views for richer training or more accurate pose evaluation, full rendering of training images for a given reflectance function, or designing of new tree questions based on surface orientation.

While we did not implement direct-global light separation, the discussion above clarifies the limitations of the current system in linking the image pixel values with the reflectance function of the object surface, in detecting channels affected by shadows, and in narrowing down the difference in pixel intensities between training and test



(a) Input image

(b) Edges extracted from image

Figure 6.3: Edge pixels extracted from an input image, illustrating that a large part of the edges only exist due to shadows. The presence of edges due to shadows considerably increase the number of voting patches, thus increasing the time taken by the voting process. Also, such edges contain little information about the object pose since the variable background shape of a bin-picking scene changes the location of those edges for the same object pose.

images due to inter-reflections, thus encouraging future research in light transport for bin-picking applications.

Bibliography

[1] P. David and D. DeMenthon, “Object recognition in high clutter images using line features,” in *IEEE International Conference on Computer Vision (ICCV)*, vol. 2, pp. 1581 –1588, 2005. 7

[2] David G. Lowe, “Three-dimensional object recognition from single two-dimensional images,” in *Artificial Intelligence*, vol. 31, pp. 355–395, 1987. 7

[3] David Lowe, “Distinctive image features from scale-invariant keypoints,” in *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004. 5, 7

[4] Ethan Rublee, Vincent Rabaud, Kurt Konolige and Gary R. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *IEEE International Conference on Computer Vision (ICCV)*, 2011. 5

[5] Vincent Lepetit and Pascal Fua, “Keypoint recognition using randomized trees,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, pp. 1465–1479, 2006. 7, 10

[6] Mustafa Ozuysal, Michael Calonder, Vincent Lepetit, and Pascal Fua, “Fast

keypoint recognition using random ferns," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, vol. 32, no. 3, pp. 448–461, 2010. 7, 10, 34, 42, 71

[7] Engin Tola, Vincent Lepetit, and Pascal Fua, "A Fast Local Descriptor for Dense Matching," in *IEEE Computer Vision and Pattern Recognition (CVPR)* 2008. 7

[8] Markus Ulrich, Christian Wiedemann, and Carsten Steger, "CAD-based recognition of 3d objects in monocular images," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1191–1198, 2009. 8, 9

[9] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit, "Gradient Response Maps for Real-Time Detection of Texture-Less Objects," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2012. 8, 9, 10, 75

[10] Carsten Steger, "Occlusion clutter, and illumination invariant object recognition," in *International Archives of Photogrammetry Remote Sensing (IAPRS)*, 2002. 9

[11] Robert Strzodka, Ivo Ihrke, and Marcus Magnor, "A graphics hardware implementation of the generalized hough transform for fast object recognition, scale, and 3d pose detection," in *In International Conference on Image Analysis and Processing (ICIAP)*, pp. 188–193, 2003. 10

[12] Vittorio Ferrari, Frederic Jurie, and Cordelia Schmid, "From images to

shape models for object detection," *International Journal of Computer Vision (IJCV)*, vol. 87, no. 3, pp. 284–303, 2010.

[13] Juergen Gall, and Victor Lempitsky, "Class-specific Hough forests for object detection," in *Decision Forests for Computer Vision and Medical Image Analysis. Advances in Computer Vision and Pattern Recognition*, pp. 143–157, 2013. 5, 10, 13, 31

[14] José Jerónimo Rodrigues, Jun-Sik Kim, Makoto Furukawa, Joao Xavier, Pedro Aguiar, and Takeo Kanade, "6D pose estimation of textureless shiny objects using random ferns for bin-picking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012. 10

[15] José Jerónimo Rodrigues, Jun-Sik Kim, Makoto Furukawa, and Takeo Kanade, "Method of recognizing a position of a workpiece from a photographed image," US patent 9361695, 2016. 10

[16] José Jerónimo Rodrigues et al., "Data-driven monocular 3D pose estimation using multi-light images", to be submitted. 10

[17] Olga Barinova, Victor Lempitsky, and Pushmeet Kohli, "On detection of multiple object instances using Hough transforms," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9), pp.1773-1784, 2012. 10, 13, 31

[18] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, Prentice Hall, Aug. 2002. 18, 22

[19] Y. Hwang, J. Kim, and I. Kweon, "Sensor noise modeling using the Skellam

distribution: Application to the color edge detection," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2007.

[20] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, pp. 932–946, 2002. 45, 69

[21] S. K. Nayar, G. Krishnan, M. D. Grossberg, and R. Raskar, "Fast Separation of Direct and Global Components of a Scene using High Frequency Illumination," in *ACM Transactions on Graphics*, vol. 25, pp. 935–944, 2006. 75, 77, 78

[22] M. Liu, O. Tuzel, A. Veeraraghavan, R. Chellappa, A. Agrawal, and H. Okuda, "Pose Estimation in Heavy Clutter using a Multi-Flash Camera," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2028–2035, 2010. 9

[23] M. Calonder, V. Lepetit, C. Strecha, P. Fua, Pascal, "BRIEF: binary robust independent elementary features," in *European Conference on Computer Vision (ECCV)*, pp. 778–792, 2010. 34

[24] J. Shotton, A. Blake, and R. Cipolla, "Multi-scale categorical object recognition using contour fragments," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, vol. 30, no. 7, pp. 1270–1281, July 2008. 9

[25] P. F. Felzenszwalb, and J. D. Schwartz, "Hierarchical matching of deformable shapes," in *CVPR 2007* Vision and Pattern Recognition, pages 1-8. 9

[26] H. Ling, and D. W. Jacobs, "Shape classification using the innerdistance," in

IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI), vol. 29, no. 2, pp. 286–299, February 2007. 9

[27] S. Nayar, X.-S. Fang, and T. Boult. "Removal of Specularities using Color and Polarization," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 1993. 9

[28] S. Mallick, T. Zickler, D. Kriegman, and P. Belhumeur, "Beyond Lambert: Reconstructing specular surfaces using color," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2005. 9

[29] Margarita Osadchy, David Jacobs, Ravi Ramamoorthi, and David Tucker, "Using specularities in comparing 3D models and 2D images," in *Computer Vision and Image Understanding*, vol. 111, no. 3, pp. 275–294, 2008. 9

[30] Aaron Netz and Margarita Osadchy, "Using specular highlights as pose invariant features for 2D-3D pose estimation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2011.

[31] Nitesh Shroff, Yuichi Taguchi, Oncel Tuzel, Ashok Veeraraghavan, Srikumar Ramalingam, and Haruhisa Okuda, "Finding a needle in a specular haystack," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5963-5970, 2011. 9

[32] Alvaro Collet, Manuel Martinez, and Siddhartha Srinivasa, "The MOPED framework: Object recognition and pose estimation for manipulation," in *International Journal of Robotics Research (IJJR)*, Sep. 2011. 5

[33] Joel Fan, and Lawrence B. Wolff, "Surface curvature and shape reconstruction from unknown multiple illumination and integrability, " in *Computer Vision and Image Understanding*, vol. 65, no. 2, pp. 347–359, 1997. 23

[34] Peter N. Belhumeur, David J. Kriegman, and Alan L. Yuille, "The Bias-relief ambiguity," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 1997. 23

[35] Steven M. Seitz, Yasuyuki Matsushita, and Kiriakos N. Kutulakos, "A theory of inverse light transport," in *IEEE International Conference on Computer Vision (ICCV)*, 2005. 77

[36] Vincent Lepetit, and Pascal Fua, "Monocular model-based 3D tracking of rigid objects: A survey," in *Foundations and Trends in Computer Graphics and Vision*, vol. 1, pp. 1–89, October 2005. 45

[37] Shree K. Nayar, and Mohit Gupta, "Diffuse structured light," in *IEEE International Conference on Computational Photography (ICCP)*, 2012. 79

[38] Jinwei Gu, Toshihiro Kobayashi, Mohit Gupta, and Shree K. Nayar, "Multiplexed illumination for scene recovery in the presence of global illumination," in *IEEE International Conference on Computer Vision (ICCV)*, pp.1-8, Nov, 2011. 75, 77, 79

[39] Yoav Y. Schechner, Shree K. Nayar, and Peter N. Belhumeur, "Multiplexing for optimal lighting," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 29, No. 8 , pp. 1339-1354, 2007. 71

[40] Yoav Y. Schechner, Shree K. Nayar, and Peter N. Belhumeur, "Theory of multiplexed illumination," in *IEEE International Conference on Computer Vision (ICCV)*, Vol. 2, pp. 808-815, 2003. 71

[41] Kai-Tai Song, Cheng-Hei Wu, and Sin-Yi Jiang, "CAD-based pose estimation design for random bin picking using a RGB-D camera," in *Journal of Intelligent and Robotic Systems*, Vol. 87, pp 455470, 2017. 10, 75

[42] Changhyun Choi, Yuichi Taguchi, Oncel Tuzel, Ming-Yu Liu, and Srikumar Ramalingam, "Voting-based pose estimation for robotic assembly using a 3D sensor," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. 10, 75

[43] Wim Abbeloos, and Toon Goedeme, "Point pair feature based object detection for random bin picking," in *IEEE Conference on Computer and Robot Vision (CRV)*, pp. 432439, 2016. 10, 75

[44] Skotheim Oystein, Jens T. Thielemann, Asbjorn Berge, and Arne Sommerfelt, "Robust 3D object localization and pose estimation for random bin picking with the 3DMaMa algorithm," in *SPIE Three-Dimensional Image Processing (3DIP) and Applications*, 75260E, 2010. 10, 75

[45] Mingyu Li, and Koichi Hashimoto, "Curve set feature-based robust and fast pose estimation algorithm," in *Sensors*, 17(8):1782, 2017. 10, 75

[46] Liang Zheng, Yi Yang, and Qi Tian, "SIFT meets CNN:'A decade survey of instance retrieval," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 71

[47] Tomasz Trzcinski, and Vincent Lepetit, "Efficient discriminative projections for compact binary descriptors," in *European Conference on Computer Vision (ECCV)*, 2012. 71

[48] Navneet Dalal, and Bill Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2005. 71

[49] Changhyun Choi, and Henrik I. Christensen, "3D textureless object detection and tracking: an edge-based approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012. 45

[50] Thomas P. Minka, "The summation hack as an outlier model," in *Tutorial Note 21*, 2003. 16, 30

[51] Alykhan Tejani, Rigas Kouskouridas, Andreas Doumanoglou, Danhang Tang, and Tae-Kyun Kim, "Latent-Class Hough Forests for 6 DoF Object Pose Estimation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 40(1), pp. 119-132, 2018. 10, 75

[52] Peter Kontschieder, Samuel Rota Bulo, Horst Bischof, and Marcello Pelillo, "Structured Class-Labels in Random Forests for Semantic Image Labelling," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 2190–2197, 2011. 75

[53] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox, "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes," in *Robotics: Science and Systems (RSS)*, 2018. 73

[54] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox, “DeepIM: Deep Iterative Matching for 6D Pose Estimation,” in *European Conference on Computer Vision (ECCV)*, 2018. 73

[55] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua, “Real-Time Seamless Single Shot 6D Object Pose Prediction”, in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018. 73

[56] Mahdi Rad, and Vincent Lepetit, “BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth,” in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 73

[57] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab, “SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again,” in *IEEE International Conference on Computer Vision (ICCV)*, 2017. 73

[58] Thanh-Toan Do, Ming Cai, Trung Pham, and Ian D. Reid, “Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image,” in *Computing Research Repository (CoRR)*, arXiv:1802.10367, 2018. 73

[59] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis, “6-DoF object pose from semantic keypoints,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 73