

Object Detection and Pose Estimation from RGB and Depth Data for Real-Time, Adaptive Robotic Grasping



Shuvo Kumar Paul, Muhammed Tawfiq Chowdhury, Mircea Nicolescu, Monica Nicolescu, and David Feil-Seifer

1 Introduction

Current advances in robotics and autonomous systems have expanded the use of robots in a wide range of robotic tasks including assembly, advanced manufacturing, human-robot or robot-robot collaboration. In order for robots to efficiently perform these tasks, they need to have the ability to adapt to the changing environment while interacting with their surroundings, and a key component of this interaction is the reliable grasping of arbitrary objects. Consequently, a recent trend in robotics research has focused on object detection and pose estimation for the purpose of dynamic robotic grasping.

However, identifying objects and recovering their poses are particularly challenging tasks as objects in the real world are extremely varied in shape and appearance. Moreover, cluttered scenes, occlusion between objects, and variance in lighting conditions make it even more difficult. Additionally, the system needs to be sufficiently fast to facilitate real-time robotic tasks. As a result, a generic solution that can address all these problems remains an open challenge.

While classification [1–6], detection [7–12], and segmentation [13–15] of objects from images have taken a significant step forward—thanks to deep learning, the same has not yet happened to 3D localization and pose estimation. One primary reason was the lack of labeled data in the past as it is not practical to manually infer, thus as a result, the recent research trend in the deep learning community for such applications has shifted towards synthetic datasets [16–20]. Several pose estimation

S. K. Paul (✉) · M. T. Chowdhury · M. Nicolescu · M. Nicolescu · D. Feil-Seifer
Department of Computer Science and Engineering, University of Nevada, Reno, NV, USA
e-mail: shuvo.k.paul@nevada.unr.edu; mtawfiqc@nevada.unr.edu; mircea@cse.unr.edu;
monica@cse.unr.edu; dave@cse.unr.edu

© Springer Nature Switzerland AG 2021

H. R. Arabnia et al. (eds.), *Advances in Computer Vision and Computational Biology*, Transactions on Computational Science and Computational Intelligence,
https://doi.org/10.1007/978-3-030-71051-4_10

121

methods leveraging deep learning techniques [21–25] use these synthetic datasets for training and have shown satisfactory accuracy.

Although synthetic data is a promising alternative, capable of generating large amounts of labeled data, it requires photorealistic 3D models of the objects to mirror the real-world scenario. Hence, generating synthetic data for each newly introduced object needs photorealistic 3D models and thus significant effort from skilled 3D artists. Furthermore, training and running deep learning models are not feasible without high computing resources as well. As a result, object detection and pose estimation in real-time with computationally moderate machines remain a challenging problem. To address these issues, we have devised a simpler pipeline that does not rely on high computing resources and focuses on planar objects, requiring only an RGB image and the depth information in order to infer real-time object detection and pose estimation.

In this work, we present a feature-detector-descriptor based method for detection and a homography based pose estimation technique where, by utilizing the depth information, we estimate the pose of an object in terms of a 2D planar representation in 3D space. The robot is pre-trained to perform a set of canonical grasps; a canonical grasp describes how a robotic end-effector should be placed relative to an object in a fixed pose so that it can securely grasp it. Afterward, the robot is able to detect objects and estimates their pose in real-time, and then adapt the pre-trained canonical grasp to the new pose of the object of interest. We demonstrate that the proposed method can detect a well-textured planar object and estimate its accurate pose within a tolerable amount of out-of-plane rotation. We also conducted experiments with the humanoid PR2 robot to show the applicability of the framework where the robot grasped objects by adapting to a range of different poses.

2 Related Work

Our work constitutes of three modules: object detection, planar pose estimation, and adaptive grasping. In the following sub-sections, several fields of research that are closely related to our work are reviewed.

2.1 Object Detection

Object detection has been one of the fundamental challenges in the field of computer vision and in that aspect, the introduction of feature detectors and descriptors represents a great achievement. Over the past decades, many detectors, descriptors, and their numerous variants have been presented in the literature. The applications of these methods have widely extended to numerous other vision applications such as panorama stitching, tracking, visual navigation, etc.

One of the first feature detectors was proposed by Harris et al. [26] (widely known as the Harris corner detector). Later Tomasi et al. [27] developed the KLT (Kanade-Lucas-Tomasi) tracker based on the Harris corner detector. Shi and Tomasi introduced a new detection metric GFTT [28] (Good Features To Track) and argued that it offered superior performance. Hall et al. introduced the concept of saliency [29] in terms of the change in scale and evaluated the Harris method proposed in [30] and the Harris Laplacian corner detector [31] where a Harris detector and a Laplacian function are combined.

Motivated by the need for a scale-invariant feature detector, in 2004 Lowe [32] published one of the most influential papers in computer vision, SIFT (Scale-Invariant Feature Transform). SIFT is both a feature point detector and descriptor. H. Bay et al. [33] proposed SURF (Speeded Up Robust Features) in 2008. But both of these methods are computationally expensive as SIFT detector leverages the difference of Gaussians (DoG) in different scales while SURF detector uses a Haar wavelet approximation of the determinant of the Hessian matrix to speed up the detection process. Many variants of SIFT [34–37] and SURF [38–40] were proposed, either targeting a different problem or reporting improvements in matching, however, the execution time remained a persisting problem for several vision applications.

To improve execution time, several other detectors such as FAST [41] and AGAST [42] have been introduced. Calonder et al. developed the BRIEF [43] (Binary Robust Independent Elementary Features) descriptor of binary strings that has a fast execution time and is very useful for matching images. E. Rublee et al. presented ORB [44] (Oriented FAST and Rotated Brief) which is a combination of modified FAST (Features from Accelerated Segment Test) for feature detection and BRIEF for description. S. Leutnegger et al. designed BRISK [45] (Binary Robust Invariant Scale Keypoint) that detects corners using AGAST and filters them using FAST. On the other hand, FREAK (Fast Retina Keypoint), introduced by Alahi et al. [46], generates retinal sampling patterns using a circular sampling grid and uses a binary descriptor, formed by a one bit difference of Gaussians (DoG). Alcantarilla et al. introduced KAZE [47] features that exploit non-linear scale-space using non-linear diffusion filtering and later extended it to AKAZE [48] where they replaced it with a more computationally efficient method called FED (Fast Explicit Diffusion) [49, 50].

In our work, we have selected four methods to investigate: SIFT, SURF, FAST+BRISK, AKAZE.

2.2 *Planar Pose Estimation*

Among the many techniques in literature on pose estimation, we focus our review on those related to planar pose estimation. In recent years, planar pose estimation has been increasingly becoming popular in many fields, such as robotics and augmented reality.

Simon et al. [51] proposed a pose estimation technique for planar structures using homography projection and by computing camera pose from consecutive images. Changhai et al. [52] presented a method to robustly estimate 3D poses of planes by applying a weighted incremental normal estimation method that uses Bayesian inference. Donoser et al. [53] utilized the properties of Maximally Stable Extremal Regions (MSERs [54]) to construct a perspective invariant frame on the closed contour to estimate the planar pose. In our approach, we applied perspective transformation to approximate a set of corresponding points on the test image for estimating the basis vectors of the object surface and used the depth information to estimate the 3D pose by computing the normal to the planar object.

2.3 Adaptive Grasping

Designing an adaptive grasping system is challenging due to the complex nature of the shapes of objects. In early times, analytical methods were used where the system would analyze the geometric structure of the object and would try to predict suitable grasping points. Sahbani et al. [55] did an in-depth review on the existing analytical approaches for 3D object grasping. However, with the analytical approach it is difficult to compute force and not suitable for autonomous manipulation. Later, as the number of 3D models increased, numerous data driven methods were introduced that would analyze grasps in the 3D model database and then transfer to the target object. Bohg et al. [56] reviewed data driven grasping method methods where they divided the approach into three groups based on the familiarity of the object.

Kehoe et al. [57] used a candidate grasp from the candidate grasp set based on the feasibility score determined by the grasp planner. The grasps were not very accurate in situations where the objects had stable horizontal poses and were close to the width of the robot's gripper. Huebner et al. [58] also take a similar approach as they perform grasp candidate simulation. They created a sequence of grasps by approximating the shape of the objects and then computed a random grasp evaluation for each model of objects. In both works, a grasp has been chosen from a list of candidate grasps (Fig. 1).

The recent advances in deep learning also made it possible to regress grasp configuration through deep convolutional networks. A number of deep learning-based methods were reviewed in [59] where the authors also discussed how each element in deep learning-based methods enhances the robotic grasping detection. [60] presented a system where deep neural networks were used to learn hierarchical features to detect and estimate the pose of an object, and then use the centers of the defined pose classes to grasps the objects. Kroemer et al. [61] introduced an active learning approach where the robot observes a few good grasps by demonstration and learns a value function for these grasps using Gaussian process regression. Aleotti et al. [62] proposed a grasping model that is capable of grasping objects by their parts which learns new tasks from human demonstration with automatic 3D shape segmentation for object recognition and semantic modeling. Saxena et al. [63]

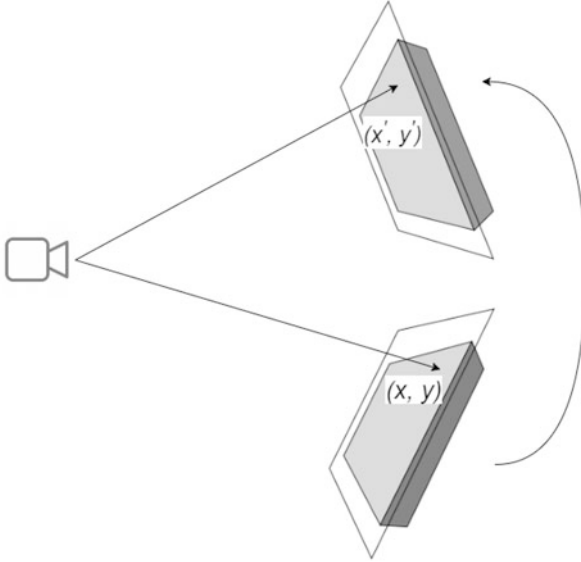


Fig. 1 Object in different orientation from the camera

and Montesano and Lopes [64] used supervised learning to predict grasp locations from RGB images. In [65], as an alternative to a trial-and-error exploration strategy, the authors proposed a Bayesian optimization technique to address the robot grasp optimization problem of unknown objects. These methods emphasized developing and using learning models for obtaining accurate grasps.

In our work, we focus on pre-defining a suitable grasp relative to an object that can adapt to a new grasp based on the change of position and orientation of the object.

3 Method

The proposed method is divided into two parts. The first part outlines the process of simultaneous object detection and pose estimation of multiple objects and the second part describes the process of generating an adaptive grasp using the pre-trained canonical grasp and the object pose. The following sections describe the architecture of the proposed framework (Fig. 2) in detail.

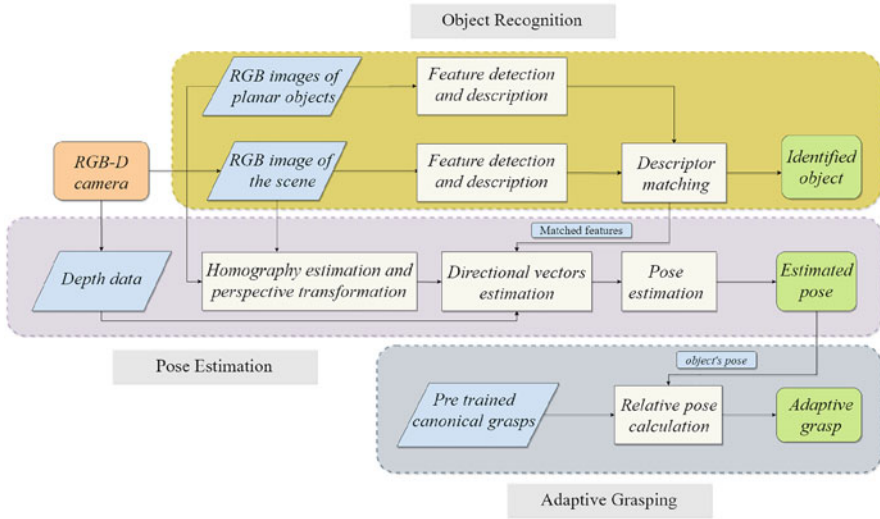


Fig. 2 System architecture

3.1 Object Detection and Pose Estimation

We present a planar pose estimation algorithm (Algorithm 1) for adaptive grasping that consists of four phases: (1) feature extraction and matching, (2) homography estimation and perspective transformation, (3) directional vectors estimation on the object surface, (4) planar pose estimation using the depth data. In the following sections, we will focus on the detailed description of the aforementioned steps.

Feature Extraction and Matching

Our object detection starts with extracting features from the images of the planar objects and then matching them with the features found in the images acquired from the camera. Image features are patterns in images based on which we can describe the image. A feature detecting algorithm takes an image and returns the locations of these patterns—they can be edges, corners or interest points, blobs or regions of interest points, ridges, etc. This feature information then needs to be transformed into a vector space using a feature descriptor, so that it gives us the possibility to execute numerical operations on them. A feature descriptor encodes these patterns into a series of numerical values that can be used to match, compare, and differentiate one feature to another; for example, we can use these feature vectors to find the similarities in different images which can lead us to detect objects in the image. In theory, this information would be invariant to image transformations. In our work, we have investigated SIFT [32], SURF [33], AKAZE [48], and

Algorithm 1: Planar pose estimation

```

Input: Training images of planar objects,  $\mathcal{I}$ 
1 Detector  $\leftarrow$  Define feature detector
2 Descriptor  $\leftarrow$  Define feature descriptor
   /* retrieve feature descriptor */
   /* for each image in  $\mathcal{I}$  */
3 for  $i$  in  $\mathcal{I}$  do
   /*  $\mathcal{K}$  is set of detected keypoints for image  $i$  */
4    $\mathcal{K} \leftarrow$  DetectKeypoints( $i$ , Detector)
   /*  $\mathcal{D}[i]$  is the corresponding descriptor set for image  $i$  */
5    $\mathcal{D}[i] \leftarrow$  GetDescriptors( $\mathcal{K}$ , Descriptor)
6 end for
7 while camera is on do
8    $f \leftarrow$  RGB image frame
9    $PC \leftarrow$  Point cloud data
   /*  $K_F$  is set of detected keypoints for image frame  $f$  */
10   $K_F \leftarrow$  DetectKeypoints( $f$ , Detector)
   /*  $D_F$  is the corresponding descriptor set for rgb image  $f$  */
11   $D_F \leftarrow$  GetDescriptors( $K_F$ , Descriptor)
12  for  $i$  in  $\mathcal{I}$  do
13     $matches \leftarrow$  FindMatches( $\mathcal{D}[i]$ ,  $D_F$ )
   /* If there is at least 10 matches then we have the object
      (described in image  $i$ ) in the scene */
14    if Total number of matches  $\geq 10$  then
   /* extract matched keypoints pair ( $kp_i, kp_f$ ) from the
      corresponding descriptors matches. */
15     $kp_i, kp_f \leftarrow$  ExtractKeypoints( $matches$ )
16     $\mathbf{H} \leftarrow$  EstimateHomography( $kp_i, kp_f$ )
17     $p_c, p_x, p_y \leftarrow$  points on the planar object
      obtained using Eq. (3)
18     $p'_c, p'_x, p'_y \leftarrow$  corresponding projected points
      of  $p_c, p_x, p_y$  on image frame  $f$ 
      estimated using equations
      (1) and (2)
   /*  $\bar{\mathbf{c}}$  denotes the origin of the object frame with respect to the
      base/world frame */
19     $\mathbf{c}, \mathbf{x}, \mathbf{y} \leftarrow$  corresponding 3d locations
      of  $p'_c, p'_x, p'_y$  from point cloud  $PC$ 
   /* shift  $\mathbf{x}, \mathbf{y}$  to the origin of the base or the world frame */
20     $\mathbf{x} \leftarrow \mathbf{x} - \mathbf{c}$ 
21     $\mathbf{y} \leftarrow \mathbf{y} - \mathbf{c}$ 
   /* estimate the object frame in terms of three orthonormal
      vectors  $\hat{i}, \hat{j}$ , and  $\hat{k}$ . */
22     $\hat{i}, \hat{j}, \hat{k} \leftarrow$  from Eq. (4)
   /* compute the rotation  $\phi_i, \theta_i, \psi_i$  of the object frame  $\hat{i}, \hat{j}, \hat{k}$  with
      respect to the base or the world frame  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ . */
23     $\phi_i, \theta_i, \psi_i \leftarrow$  from Eq. (8)
   /* finally, publish the position and orientation of the object.
      */
24    publish( $\mathbf{c}, \phi_i, \theta_i, \psi_i$ )
25  end for
26 end while

```

BRISK [45] descriptors. SIFT, SURF, AKAZE are both feature detectors and descriptors, but BRISK uses FAST [41] algorithm for feature detection. These descriptors were selected after carefully reviewing the comparisons done in the recent literature [66, 67].

Once the features are extracted and transformed into vectors, we compare the features to determine the presence of an object in the scene. For non-binary feature descriptors (SIFT, SURF) we find matches using the Nearest Neighbor algorithm. However, finding the nearest neighbor matches within high dimensional data is computationally expensive, and with more objects introduced it can affect the process of updating the pose in real-time. To counter this issue to some extent, we used the FLANN [68] implementation of K-d Nearest Neighbor Search, which is an approximation of the K-Nearest Neighbor algorithm that is optimized for high dimensional features. For binary features (AKAZE, BRISK), we used the Hamming distance ratio method to find the matches. Finally, if we have more than ten matches, we presume the object is present in the scene.

Homography Estimation and Perspective Transformation

A homography is an invertible mapping of points and lines on the projective plane that describes a 2D planar projective transformation (Fig. 1) that can be estimated from a given pair of images. In simple terms, a homography is a matrix that maps a set of points in one image to the corresponding set of points in another image. We can use a homography matrix \mathbf{H} to find the corresponding points using Eqs. (1) and (2), which defines the relation of projected point (x', y') (Fig. 1) on the rotated plane to the reference point (x, y) .

A 2D point (x, y) in an image can be represented as a 3D vector $(x, y, 1)$ which is called the homogeneous representation of a point that lies on the reference plane or image of the planar object. In Eq. (1), \mathbf{H} represents the homography matrix and $[x \ y \ 1]^T$ is the homogeneous representation of the reference point (x, y) and we can use the values of a, b, c to estimate the projected point (x', y') in Eq. (2).

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (1)$$

$$\begin{cases} x' = \frac{a}{c} \\ y' = \frac{b}{c} \end{cases} \quad (2)$$

We estimate the homography using the matches found from the nearest neighbor search as input; often these matches can have completely false correspondences, meaning they do not correspond to the same real-world feature at all which can be

a problem in estimating the homography. So, we chose RANSAC [69] to robustly estimate the homography by considering only inlier matches as it tries to estimate the underlying model parameters and detect outliers by generating candidate solutions through random sampling using a minimum number of observations.

While the other techniques use as much data as possible to find the model parameters and then pruning the outliers, RANSAC uses the smallest set of data point possible to estimate the model, thus making it faster and more efficient than the conventional solutions.

Finding Directional Vectors on the Object

In order to find the pose of a planar object, we need to find the three orthonormal vectors on the planar object that describe the object coordinate frame and consequently, the orientation of the object relative to the world coordinate system. We start by estimating the vectors on the planar object that form the basis of the plane, illustrated in Fig. 3. Then, we take the cross product of these two vectors to find the third directional vector which is the normal to the object surface. Let us denote the world coordinate system as XYZ , and the object coordinate system as xyz . We define the axes of the orientation in relation to a body as:

$$\begin{aligned} x &\rightarrow \text{right} \\ y &\rightarrow \text{up} \\ z &\rightarrow \text{towards the camera} \end{aligned}$$

First, we retrieve the locations of the three points p_c, p_x, p_y on the planar object from the reference image using Eq. (3) and then locate the corresponding points p'_c, p'_x, p'_y on the image acquired from the Microsoft Kinect sensor. We estimate the locations of these points using the homography matrix \mathbf{H} as shown in Eqs. (1) and (2). Then we find the corresponding 3D locations of p'_c, p'_x, p'_y from the point cloud data also obtained from the Microsoft Kinect sensor. We denote them as vectors \mathbf{c}, \mathbf{x} , and \mathbf{y} . Here, \mathbf{c} represents the translation vector from the object frame to the world frame and also the position of the object in the world frame. Next, we subtract \mathbf{c} from \mathbf{x}, \mathbf{y} which essentially gives us two vectors \mathbf{x} and \mathbf{y} centered at the origin of the world frame. We take the cross product of these two vectors \mathbf{x}, \mathbf{y} to find the third axis \mathbf{z} . But, depending on the homography matrix the estimated axes \mathbf{x} and \mathbf{y} might not be exactly orthogonal, so we take the cross product of \mathbf{y} and \mathbf{z} to recalculate the vector \mathbf{x} . Now that we have three orthogonal vectors, we compute the three unit vectors \hat{i}, \hat{j} , and \hat{k} along the \mathbf{x}, \mathbf{y} , and \mathbf{z} vectors, respectively, using Eq. (4). These three orthonormal vectors describe the object frame. These vectors were projected onto the image plane to give a visual confirmation of the methods applied; Fig. 4 shows the orthogonal axes projected onto the object plane.

Fig. 3 Axis on the reference plane

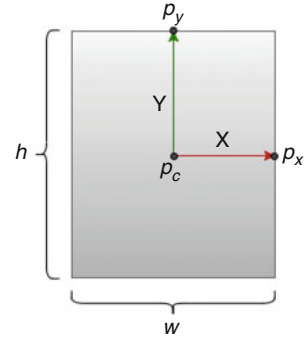


Fig. 4 Computed third directional axis projected onto image plane

$$\begin{cases} p_c = (w/2, h/2) \\ p_x = (w, h/2) \\ p_y = (w/2, 0) \end{cases} \quad (3)$$

$$\begin{aligned} \hat{j} &= \frac{\mathbf{y}}{|\mathbf{y}|} = [j_X \ j_Y \ j_Z] \\ \hat{k} &= \frac{\mathbf{x} \times \mathbf{y}}{|\mathbf{x} \times \mathbf{y}|} = [k_X \ k_Y \ k_Z] \\ \hat{i} &= \frac{\mathbf{y} \times \mathbf{z}}{|\mathbf{y} \times \mathbf{z}|} = [i_X \ i_Y \ i_Z] \end{aligned} \quad (4)$$

Planar Pose Computation

We compute the pose of the object in terms of the Euler angles. Euler angles are three angles that describe the orientation of a rigid body with respect to a fixed coordinate system. The rotation matrix \mathbf{R} in Eq. (5) rotates X axis to \hat{i} , Y axis to \hat{j} , and Z axis to \hat{k} .

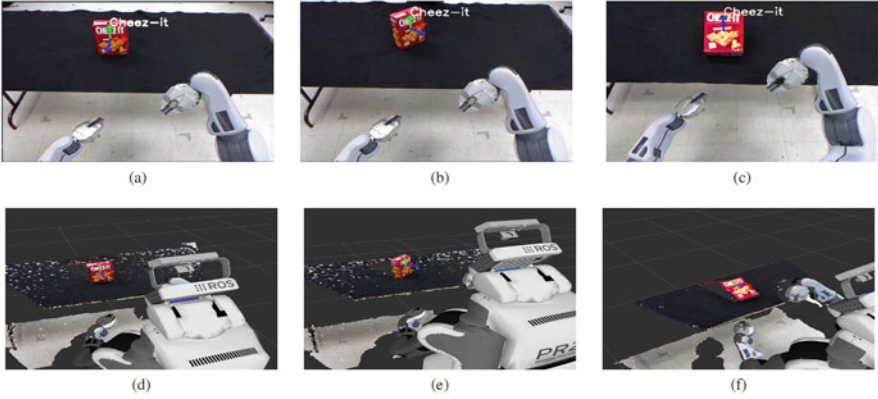


Fig. 5 (a), (b), (c) are recovered poses from robot's camera and (d), (e), (f) are corresponding poses visualized in RViz

$$\mathbf{R} = \begin{bmatrix} i_X & j_X & k_X \\ i_Y & j_Y & k_Y \\ i_Z & j_Z & k_Z \end{bmatrix} \quad (5)$$

Euler angles are combinations of the three axis rotations (Eq. (6)), where ϕ , θ , and ψ specify the intrinsic rotations around the X, Y, and Z axis, respectively. The combined rotation matrix is a product of three matrices: $\mathbf{R} = \mathbf{R}_z \mathbf{R}_y \mathbf{R}_x$ (Eq. (7)); the first intrinsic rotation rightmost, last leftmost (Fig. 5).

$$\left\{ \begin{array}{l} \mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix} \\ \mathbf{R}_y = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \\ \mathbf{R}_z = \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array} \right. \quad (6)$$

$$\mathbf{R} = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \quad (7)$$

In Eq. (7), c and s represents \cos and \sin , respectively.

Solving for ϕ , θ , and ψ from (5) and (7), we get,

$$\begin{cases} \phi = \tan^{-1} \left(\frac{j_Z}{k_Z} \right) \\ \theta = \tan^{-1} \left(\frac{-i_Z}{\sqrt{1 - i_Z^2}} \right) = \sin^{-1} (-i_Z) \\ \psi = \tan^{-1} \left(\frac{i_Y}{i_X} \right) \end{cases} \quad (8)$$

3.2 Training Grasps for Humanoid Robots

To ensure that the robot can grasp objects in an adaptive manner, we pre-train the robot to perform a set of canonical grasps. We place the object and the robot's gripper close to each other and record the relative pose. This essentially gives us the pose of the gripper with respect to the object. Figure 6 illustrates the training process in which the robot's gripper and a cracker box have been placed in close proximity and the relative poses have been recorded for grasping the objects from the side.



Fig. 6 Pre-training canonical grasp

$$\mathbf{T}_s^d = \begin{bmatrix} \mathbf{R}_s^d & P_s^d \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & X_t \\ r_{21} & r_{22} & r_{23} & Y_t \\ r_{31} & r_{32} & r_{33} & Z_t \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

Equation (9) outlines the structure of a transformation matrix \mathbf{T}_s^d that describes the rotation and translation of frame d with respect to frame s ; \mathbf{R}_s^d represents the rotation matrix similar to Eq. (7) and $P_s^d = [X_t, Y_t, Z_t]^T$ is the translation matrix which is the 3D location of the origin of frame d in frame s .

During the training phase, we first formulate the transformation matrix \mathbf{T}_b^o using the rotation matrix and the object location. We take the inverse of \mathbf{T}_b^o which gives us the transformation matrix \mathbf{T}_o^b . We then use the Eq. (10) to record the transformation \mathbf{T}_o^g of the robot's wrist relative to the object.

$$T_o^g = T_o^b \times T_b^g \text{ where } T_o^b = (T_b^o)^{-1} \quad (10)$$

In the Eq. (10), b refers to the robot's base, o refers to the object, and g refers to the wrist of the robot to which the gripper is attached. Once we record the matrix, we get a new pose of the object from the vision in the testing phase and generate the final matrix using the Eq. (11) that has the new position and orientation of the robot's wrist in matrix form .

$$T_b^g = T_b^o \times T_o^g \quad (11)$$

We then extract the rotational angles γ , β , α (roll, pitch, yaw) of the grasp pose from matrix \mathbf{T}_b^g using Eq. (12)

$$\begin{cases} \gamma = \tan^{-1}(r_{32}/r_{33}) \\ \beta = \tan^{-1} \frac{-r_{31}}{\sqrt{r_{32}^2 + r_{33}^2}} \\ \alpha = \tan^{-1}(r_{21}/r_{11}) \end{cases} \quad (12)$$

4 Evaluation

The proposed object recognition and pose estimation algorithm was implemented on an Ubuntu 14.04 platform equipped with 3.0 GHz Intel R Core(TM) i5-7400 CPU and 8GB system memory. The RGB-D camera used in the experiments was a Microsoft Kinect sensor v1. We evaluated the proposed algorithm by comparing the accuracy of object recognition, pose estimation, and execution time of four different feature descriptors. We also validated the effectiveness of our approach for adaptive grasping by conducting experiments with the PR2 robot.

4.1 Object Detection and Pose Estimation

Without enough observable features, the system would fail to find good matches that are required for accurate homography estimation. Consequently, our object detection and pose estimation approach has a constraint on the out-of-plane rotation θ , illustrated in Fig. 7. In other words, if the out-of-plane rotation of the object is more than θ , the system would not be able to recognize the object. Fast execution is also a crucial aspect to facilitate multiple object detection and pose estimation for real-time applications. We experimented with four different descriptors on several planar objects and the comparative result is shown in Table 1. The execution time was measured for the object detection and pose estimation step. AKAZE and BRISK had much lower processing time for detection and pose estimation, thus would have a better frame rate, but SIFT and SURF had larger out-of-plane rotational freedom.

We also compared the *RMS* difference ϵ (Eq. (13)) of re-calculated \mathbf{x} to original \mathbf{x} (\mathbf{x}' in the equation) for increasing out-of-plane rotation of the planar objects to assess the homography estimation. Ideally, the two estimated vectors \mathbf{x} and \mathbf{y} , which

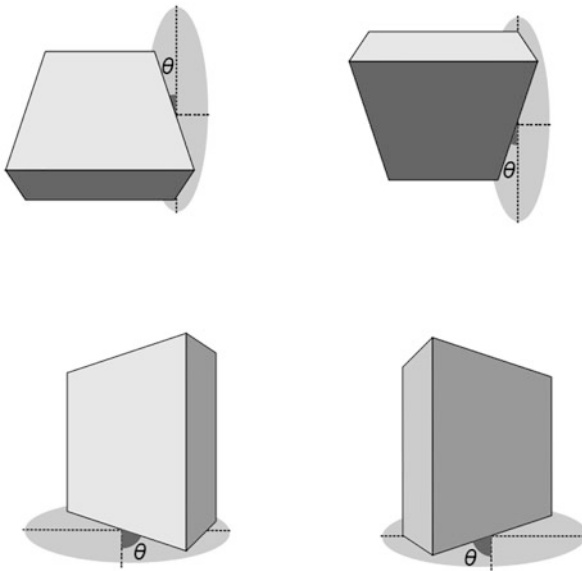


Fig. 7 Out-of-plane rotation

Table 1 Comparison of feature descriptors

Descriptor	Maximum out-of-plane rotation (degree)	Execution time (second)
SIFT	$48 \pm 2^\circ$	0.21 s
SURF	$37 \pm 2^\circ$	0.27 s
AKAZE	$18 \pm 1^\circ$	0.05 s
BRISK	$22 \pm 2^\circ$	0.06 s

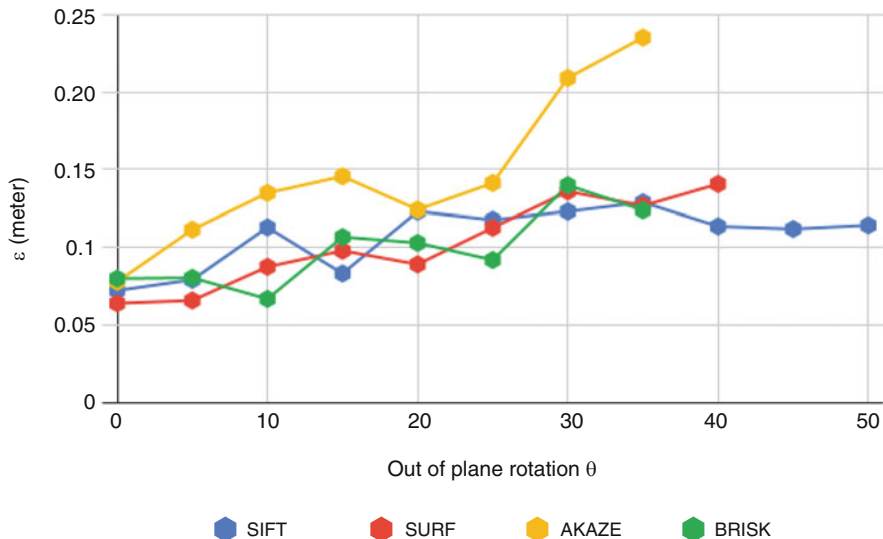


Fig. 8 Out-of-plane rotation vs ϵ

Table 2 Execution time of SIFT and SURF for multiple object detection

Number of Objects	Detection time (second)	
	SIFT	SURF
1	0.06 s	0.09 s
2	0.11 s	0.17 s
3	0.17 s	0.26 s
4	0.22 s	0.35 s
5	0.28 s	0.45 s
6	0.34 s	0.54 s

describe the basis of the plane of the planar object, should be orthogonal to each other, but often they are not. So, the values of ϵ in Fig. 8 give us an indication of the average error in homography estimation for different out-of-plane rotations. In Fig. 8, we can see AKAZE has much higher ϵ values while the rest remained within a close range. This tells us AKAZE results in a much larger error in estimating the homography than the other methods.

We chose SIFT and SURF to evaluate how the execution time for detection scales up while increasing the number of objects. From Table 2, which shows the mean processing time for object detection, we can see that SURF had a detection time around 50% more than SIFT in all the cases. This outcome coupled with the previous results prompted us to select SIFT for the subsequent experiments.

The system was capable of detecting multiple objects in real-time and at the same time could estimate their corresponding poses. Figure 9 shows detected objects with estimated directional planar vectors. We can also observe that the system was robust to in-plane rotation and partial occlusion.



Fig. 9 Multiple object detection with estimated planar vectors



Fig. 10 (a) Pose estimation of multiple objects (b) Estimated pose of an object held by a human

We used RViz, a 3D visualizer for the Robot Operating System (ROS), to validate the pose estimation. The calculated directional axes were projected onto the image and the estimated poses were visualized in RViz. As shown in Fig. 5, we qualitatively verified the accuracy of the detection and the estimated pose by comparing the two outputs. We can see that both the outputs render similar results. We conducted experiments with multiple objects and human held objects as well. Figure 10 illustrates the simultaneous detection and pose estimation of two different boxes and an object held by a human, respectively.

$$\epsilon = \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}'_i - \mathbf{x}_i\|, \text{ where } N \text{ is the number of frames} \quad (13)$$

4.2 Adaptive Grasping

We assessed our approach for adaptive grasping keeping two different aspects of the robotic application in mind; robotic tasks that require (1) interacting with a static environment, and (2) interacting with humans.

We first tested our system for static objects where the object was attached to a tripod. Next, we set up experiments where the object was held by a human. We used a sticker book and a cartoon book and evaluated our system on a comprehensive set of poses. In almost all the experiments, the robot successfully grasped the object in a manner consistent with its training. There were some poses that were not reachable by the robot—for instance, when the object was pointing inward along the X axis in the robot reference frame, it was not possible for the end-effector to make a top grasp. Figures 11 and 12 show the successful grasping of the robot for both types of experiments.



Fig. 11 Robot grasping an object from a tripod. Left: initial position of the robot's gripper, middle: gripper adapting to the object's pose, right: grasping of the object



Fig. 12 Robot grasping an object held by a human. Left: initial position of the robot's gripper, middle: gripper adapting to the object's pose, right: grasping of the object

5 Conclusion and Future Work

This work presents an approach that enables humanoid robots to grasp objects using planar pose estimation based on RGB image and depth data. We examined the performance of four feature-detector-descriptors for object recognition and found SIFT to be the best solution. We used FLANN's K-d Tree Nearest Neighbor implementation, and Bruteforce Hamming to find the keypoint matches and employed RANSAC to estimate the homography. The homography matrix was

used to approximate the three orthonormal directional vectors on the planar object using perspective transformation. The pose of the planar object was estimated from the three directional vectors. The system was able to detect multiple objects and estimate the pose of the objects in real-time. We also conducted experiments with the humanoid PR2 robot to show the practical applicability of the framework where the robot grasped objects by adapting to a range of different poses.

In the future, we plan to add GPU acceleration for the proposed algorithm that would further improve the overall computational efficiency of the system. We would like to extend the algorithm to automatically prioritize certain objects and limit the number of objects needed for detection based on different scheduled tasks. Finally, we would like to incorporate transferring grasp configuration for familiar objects and explore other feature matching technique, e.g., multi probe LSH, hierarchical k-means tree, etc.

Acknowledgments This work has been supported in part by the Office of Naval Research award N00014-16-1-2312 and US Army Research Laboratory (ARO) award W911NF-20-2-0084.

References

1. K. He, et al., Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778
2. S. Liu, W. Deng, Very deep convolutional neural network based image classification using small training sample size, in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)* (2015), pp. 730–734
3. C. Szegedy, et al., Going deeper with convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 1–9
4. D.C. Ciresan, et al., Flexible, high performance convolutional neural networks for image classification, in *Twenty-Second International Joint Conference on Artificial Intelligence* (2011)
5. P. Sermanet, et al., Overfeat: integrated recognition, localization and detection using convolutional networks, in *2nd International Conference on Learning Representations (ICLR 2014), Conference Date: 14-04-2014 Through 16-04-2014* (2014)
6. K. He, et al., Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(9), 1904–1916 (2015)
7. R. Girshick, Fast R-CNN, in *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1440–1448
8. S. Ren, et al., Faster R-CNN: towards real-time object detection with region proposal networks, in *Advances in Neural Information Processing Systems* (2015), pp. 91–99
9. W. Liu, et al., SSD: single shot multibox detector, in *European Conference on Computer Vision* (Springer, Berlin, 2016), pp. 21–37
10. J. Redmon, et al., You only look once: unified, real-time object detection, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 779–788
11. J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 7263–7271
12. T.-Y. Lin, et al., Focal loss for dense object detection, in *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 2980–2988
13. V. Badrinarayanan, et al., Segnet: a deep convolutional encoder-decoder architecture for image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(12), 2481–2495 (2017)

14. K. He, et al., Mask R-CNN, in *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 2961–2969
15. O. Ronneberger, et al., U-net: convolutional networks for biomedical image segmentation, in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Springer, Berlin, 2015), pp. 234–241
16. D.J. Butler, et al., A naturalistic open source movie for optical flow evaluation, in *Computer Vision – ECCV 2012*, ed. by A. Fitzgibbon et al. (Springer, Berlin, 2012), pp. 611–625
17. N. Mayer, et al., A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4040–4048
18. W. Qiu, A. Yuille, Unrealcv: connecting computer vision to unreal engine, in *European Conference on Computer Vision* (Springer, Berlin, 2016), pp. 909–916
19. Y. Zhang, et al., Unrealstereo: a synthetic dataset for analyzing stereo vision (2016, preprint). arXiv:1612.04647
20. J. McCormac, et al., Scenenet RGB-D: can 5m synthetic images beat generic imagenet pre-training on indoor segmentation? in *The IEEE International Conference on Computer Vision (ICCV)* (2017)
21. Y. Xiang, et al., Posecnn: a convolutional neural network for 6d object pose estimation in cluttered scenes, in *Robotics: Science and Systems (RSS)* (2018)
22. J. Tremblay, et al., Deep object pose estimation for semantic robotic grasping of household objects, in *Conference on Robot Learning (CoRL)* (2018)
23. E. Brachmann, et al., Learning 6d object pose estimation using 3d object coordinates, in *European Conference on Computer Vision* (Springer, Berlin, 2014), pp. 536–551
24. C. Wang, et al., Densefusion: 6d object pose estimation by iterative dense fusion, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 3343–3352
25. Y. Hu, et al., Segmentation-driven 6d object pose estimation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 3385–3394
26. C.G. Harris, et al., A combined corner and edge detector, in *Alvey Vision Conference*, vol. 15 (Citeseer, 1988), pp. 10–5244
27. C. Tomasi, T. Kanade, Detection and tracking of point features. School of Computer Science, Carnegie Mellon University, Pittsburgh (1991)
28. J. Shi, et al., Good features to track, in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, Piscataway, 1994), pp. 593–600
29. D. Hall, et al., Saliency of interest points under scale changes, in *British Machine Vision Conference (BMVC)* (2002), pp. 1–10
30. T. Lindeberg, Feature detection with automatic scale selection. *Int. J. Comput. Vis.* **30**(2), 79–116 (1998)
31. K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, in *Proceedings Eighth IEEE International Conference on Computer Vision (ICCV 2001)*, vol. 1 (IEEE, Piscataway, 2001), pp. 525–531
32. D.G. Lowe, Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**, 91–110 (2004)
33. H. Bay, et al., SURF: speeded up robust features, in *Computer Vision – ECCV 2006*, ed. by A. Leonardis, et al. (Springer Berlin, 2006), pp. 404–417
34. Y. Ke, R. Sukthankar, PCA-sift: a more distinctive representation for local image descriptors, in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2004)*, vol. 2 (IEEE, Piscataway, 2004), p. II
35. S.K. Lodha, Y. Xiao, GSIFT: geometric scale invariant feature transform for terrain data, in *Vision Geometry XIV*, vol. 6066 (International Society for Optics and Photonics, Bellingham, 2006), p. 60660L
36. A.E. Abdel-Hakim, A.A. Farag, CSIFT: a sift descriptor with color invariant characteristics, in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06)*, vol. 2 (IEEE, Piscataway, 2006), pp. 1978–1983

37. J.-M. Morel, G. Yu, ASIFT: a new framework for fully affine invariant image comparison. *SIAM J. Imag. Sci.* **2**(2), 438–469 (2009)
38. P.F. Alcantarilla, et al., Gauge-surf descriptors. *Image Vis. Comput.* **31**(1), 103–116 (2013)
39. T.-K. Kang, et al., MDGHM-surf: a robust local image descriptor based on modified discrete Gaussian–Hermite moment. *Pattern Recognit.* **48**(3), 670–684 (2015)
40. J. Fu, et al., C-surf: colored speeded up robust features, in *International Conference on Trustworthy Computing and Services* (Springer, Berlin, 2012), pp. 203–210
41. E. Rosten, T. Drummond, Machine learning for high-speed corner detection, in *European Conference on Computer Vision* (Springer, Berlin, 2006), pp. 430–443
42. E. Mair, et al., Adaptive and generic corner detection based on the accelerated segment test, in *European Conference on Computer Vision* (Springer, Berlin, 2010), pp. 183–196
43. M. Calonder, et al., Brief: computing a local binary descriptor very fast. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(7), 1281–1298 (2011)
44. E. Rublee, et al., ORB: an efficient alternative to sift or surf, in *2011 International Conference on Computer Vision* (2011), pp. 2564–2571
45. S. Leutenegger, et al., Brisk: binary robust invariant scalable keypoints, in *2011 International Conference on Computer Vision* (IEEE, Piscataway, 2011), pp. 2548–2555
46. R. Ortiz, Freak: fast retina keypoint, in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), CVPR'12, Washington* (IEEE Computer Society, Washington, 2012), pp. 510–517
47. P.F. Alcantarilla, et al., Kaze features, in *Computer Vision – ECCV 2012*, ed. by A. Fitzgibbon, et al. (Springer, Berlin, 2012), pp. 214–227
48. P.F. Alcantarilla, et al., Fast explicit diffusion for accelerated features in nonlinear scale spaces, in *British Machine Vision Conference (BMVC)* (2013)
49. J. Weickert, et al., Cyclic schemes for PDE-based image analysis. *Int. J. Comput. Vis.* **118**(3), 275–299 (2016)
50. S. Grewenig, et al., From box filtering to fast explicit diffusion, in *Joint Pattern Recognition Symposium* (Springer, Berlin, 2010), pp. 533–542
51. G. Simon, M. Berger, Pose estimation for planar structures. *IEEE Comput. Graph. Appl.* **22**(6), 46–53 (2002)
52. C. Xu, et al., 3D pose estimation for planes, in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops* (2009), pp. 673–680
53. M. Donoser, et al., Robust planar target tracking and pose estimation from a single concavity, in *2011 10th IEEE International Symposium on Mixed and Augmented Reality* (2011), pp. 9–15
54. D. Nistér, H. Stewénus, Linear time maximally stable extremal regions, in *Computer Vision – ECCV 2008*, ed. by D. Forsyth et al. (Springer, Berlin, 2008), pp. 183–196
55. A. Sahbani, et al., An overview of 3d object grasp synthesis algorithms. *Robot. Auton. Syst.* **60**(3), 326–336 (2012)
56. J. Bohg, et al., Data-driven grasp synthesis—a survey. *IEEE Trans. Robot.* **30**(2), 289–309 (2013)
57. B. Kehoe, et al., Cloud-based robot grasping with the google object recognition engine, in *2013 IEEE International Conference on Robotics and Automation* (IEEE, Piscataway, 2013)
58. K. Huebner, et al., Minimum volume bounding box decomposition for shape approximation in robot grasping, in *2008 IEEE International Conference on Robotics and Automation* (IEEE, Piscataway, 2008)
59. S. Caldera, et al., Review of deep learning methods in robotic grasp detection. *Multimodal Technol. Interact.* **2**(3), 57 (2018)
60. J. Yu, et al., A vision-based robotic grasping system using deep learning for 3d object recognition and pose estimation, in *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (IEEE, Piscataway, 2013)
61. O. Kroemer, et al., Active learning using mean shift optimization for robot grasping, in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, Piscataway, 2009)

62. J. Aleotti, S. Caselli, Part-based robot grasp planning from human demonstration, in *2011 IEEE International Conference on Robotics and Automation* (IEEE, Piscataway, 2011)
63. A. Saxena, et al., Robotic grasping of novel objects using vision. *Int. J. Robot. Res.* **27**(2), 157–173 (2008)
64. L. Montesano, M. Lopes, Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions. *Robot. Auton. Syst.* **60**(3), 452–462 (2012)
65. J. Nogueira, et al., Unscented Bayesian optimization for safe robot grasping, in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, Piscataway, 2016)
66. E. Karami, et al., Image matching using sift, surf, brief and orb: performance comparison for distorted images, in *The 24th Annual Newfoundland Electrical and Computer Engineering Conference, NECEC* (2015)
67. S.A.K. Tareen, Z. Saleem, A comparative analysis of sift, surf, kaze, akaze, orb, and brisk, in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)* (IEEE, Piscataway, 2018), pp. 1–10
68. M. Muja, D.G. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, in *International Conference on Computer Vision Theory and Application VISSAPP'09* (INSTICC Press, Lisboa, 2009), pp. 331–340
69. M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM* **24**(6), 381–395 (1981)