Interactive Segmentation, Tracking, and Kinematic Modeling of Unknown Articulated Objects

Dov Katz
Moslem Kazemi
J. Andrew Bagnell
Anthony Stentz

CMU-RI-TR-12-06

March 2012

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

We present an interactive perceptual skill for segmenting, tracking, and kinematic modeling of 3D articulated objects. This skill is a prerequisite for general manipulation in unstructured environments. Robot-environment interaction is used to move an unknown object, creating a perceptual signal that reveals the kinematic properties of the object. The resulting perceptual information can then inform and facilitate further manipulation. The algorithm is computationally efficient, handles occlusion, and depends on little object motion; it only requires sufficient texture for visual feature tracking. We conducted experiments with everyday objects on a mobile manipulation platform equipped with an RGB-D sensor. The results demonstrate the robustness of the proposed method to lighting conditions, object appearance, size, structure, and configuration.

Contents

1 Introduction

We propose an interactive perceptual skill for the manipulation of unknown objects that possess inherent degrees of freedom (see Fig. 1). This category of objects—rigid articulated objects—includes many everyday objects ranging from pliers, scissors and staplers, to windows, doors, drawers, books, and toys. To manipulate an articulated object, the robot must be able to deliberately change its configuration. For example, to manipulate a door, the degree of freedom (hinge) connecting the door to its frame is actuated.

In this paper, we develop the necessary interactive perceptual capabilities to segment an object out of its cluttered background, track the object's position and configuration over time, and model its kinematic structure. These capabilities are fundamental for purposeful manipulation.

determine its kinematic structure. It is also difficult to manipulate a complex object without continuous feedback. Thus, we argue that perception and manipulation cannot be separated. We propose to develop adequate capabilities for autonomous manipulation by closely integrating action and perception. We refer to this process as "Interactive Perception" (Fig. 2).

We contend that perception and manipulation fundamentally cannot be separated: it is difficult, if at all possible, to visually segment an unknown object and determine its kinematic structure; equally, it is difficult to manipulate a complex object without continuous sensing feedback. We refer to the combined process as "Interactive Perception" (Fig. 2). In interactive perception, the robot manipulates the environment in order to assist the perceptual process. Perception, in turn, provides information necessary for successful manipulation. Interactive perception enables the robot to perceive the outcome of its interactions, and therefore to interpret the sensor stream within the context of its actions.

In the following, we rely on interactive perception to generate relative motion, a strong visual signal. This relative motion reveals properties of the environment that would otherwise remain hidden. In our case, interaction reveals the degree of freedom and shape of objects.

Our perceptual skill requires an RGB-D sensor. This sensor co-registers color and depth images (e.g. structured light sensors or co-registered LADAR and cameras). It provides our robot with the visual information necessary to model new objects. We complement this visual information with interaction. In this paper, the robot's interactions with the environment are scripted. However, interactions can be learned from experience (e.g., see [7]). Our skill assumes no prior object models. It is insensitive to changes in lighting conditions, object size, configuration and appearance. It can model objects with an arbitrary number of degrees of freedom. It is also computationally efficient. Additionally, it allows for objects to be occluded by the manipulator during the interaction. The algorithm relies on only two assumptions: objects have enough texture to enable feature tracking, and the robot is capable of exciting the object's degrees of freedom—a reasonable assumption given that our goal is to acquire information for manipulating the object.

In the following, we describe our interactive perceptual skill in detail. In Section 2 we discuss related work, and argue that our approach outperforms prior work in speed,

Figure 1: Objects that possess inherent degrees of freedom. These degrees of freedom are closely related to the function of the object. Without prior knowledge, the degrees of freedom of an object cannot be extracted from visual information alone. They have to be discovered through interaction.

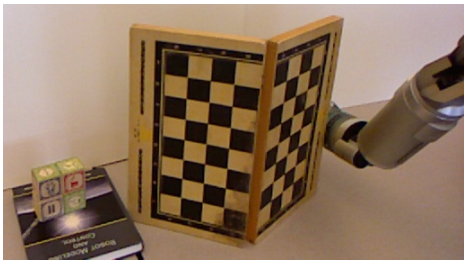


Figure 2: Our mobile manipulator interacts with an unknown articulated object (checkerboard). The proposed interactive perceptual skill segments the object, tracks its position, and models its shape and kinematic structure.

robustness, and generality. In sections 3— 6, we provide the details of our implementation. Then, in section 7, we present experiments that demonstrate our robot's ability to segment, track and model the shape and kinematic structure of everyday objects.

2 Related Work

Manipulating articulated objects is a prerequisite for a large variety of manipulation tasks. Given a priori object and environment models, robots are already capable of performing complex manipulation tasks with high precision and speed. Therefore, we believe that perception is a key obstacle for autonomous manipulation of unknown objects.

Our perceptual skill is composed of two important steps: segmenting the distinct rigid bodies in the scene, and determining the kinematic constraints between these bodies. We now review the methods and techniques that are most relevant to each of the two steps.

2.1 Object Segmentation

To manipulate an object, a robot must be able to segment it out of a cluttered scene. Indeed, image segmentation has been a focus for the computer vision community for over three decades. In image segmentation, boundaries around image regions with consistent properties are identified [5]. Existing segmentation methods typically analyze a single image, identifying discontinuities in color, brightness, texture, or depth [1, 5], with image regions computed by grouping pixels according to these properties. As objects may be multi-colored, multi-textured, and possess internal depth discontinuities, the regions computed by image segmentation methods rarely correspond to object boundaries.

Manipulation provides a grounded semantics for segmentation: an image region should correspond to a single rigid body. A rigid body is defined by the fact that the distance between any two points on the body remains constant in time regardless of external forces exerted on it. To effectively segment an unknown object we must observe the object in motion. A new generation of interactive segmentation algorithms leverages this insight that relative motion can greatly simplify image segmentation. These algorithms [4, 9] rely on the robot's body to generate motion, thereby revealing a visual signal which can be analyzed to segment the moving rigid bodies. Our approach also relies on interaction to solve the image segmentation problem.

2.2 Modeling Kinematic Constraints

The problem of acquiring kinematic models from sensor data is fundamental for autonomous manipulation, yet only recently researchers have started to address this problem. Most existing approaches assume prior knowledge about the modeled object. Our goal is manipulating unknown objects. Therefore, we focus on those methods that do not require prior object models.

Yan and Pollefeys [16] rely on structure from motion to extract 3D feature trajectories from a 2D camera, and then use spectral clustering to identify rigid bodies and their kinematic relationship. This work assumes affine geometry of the scene and only considers revolute joints. Ross et al. [12] also rely on structure from motion, but use maximum likelihood estimation to cluster features into rigid bodies. The strength of these two algorithms is that they can handle bodies that undergo slight deformation during motion. However, both approaches only handle revolute joints and make strong assumptions to simplify the perception problem. They are computationally very expensive and require large relative motions. Therefore, the above methods are not practical for autonomous manipulation.

Katz et al. [6, 7] extract kinematic models of planar articulated objects. This work relies on the insight discussed in section 2.1: deliberate interaction are used to generate relative motion, which in turn enables segmentation and joint detection. This approach is relatively computationally efficient and can handle an arbitrary number of rigid bodies and degrees of freedom, but is limited to planar objects.

Sturm et al. [13, 14] learn models of kinematic joints from three-dimensional trajectories of a moving plane. Motion is generated from deliberate interactions with the environment. The strength of this approach is that it can model arbitrary joint types.

However, it requires that only a single rigid body is moving at a time. It is also limited to objects that can be approximated as a moving plane, such as drawers and doors.

The above mentioned interactive perception approach [6, 7] has recently been extended to handle 3D objects [8]. This approach applies to general articulated objects. Similar to our approach, it requires that objects have sufficient texture for feature tracking. However, it relies on structure from motion, and therefore depends on large relative motions. It is also prohibitively slow for autonomous manipulation, averaging 15 minutes per experiment, and, similar to all of the above, it does not handle occlusion during interaction.

In contrast with the above methods, we propose an efficient solution that is practical for autonomous manipulation. In our experiments, the average runtime was 20 seconds, an order of magnitude faster than the performance of [8]. Furthermore, our method can model objects in the presence of occlusion during interaction—an important property as interaction for perception is unlikely to be gentle and precise. Finally, we acquires 3D measurements from the sensor. We do not need to reconstruct depth from motion, and therefore require little motion to segment and model unknown objects (Fig. 5).

3 Perceiving Unknown Objects

Successful manipulation of an articulated object requires the perceptual capabilities to detect and segment the object, model the kinematic constraints between its parts, and track the configuration of the object over time. These perceptual capabilities cannot be achieved from visual inspection alone, and would be extremely difficult to determine by manipulation. We now describe a perceptual skill which combines interaction with perception to achieve these capabilities.

The proposed skill is composed of three steps. The first step collects perceptual information that provides the input to our algorithm. In this step, we initialize and track visual features throughout the robot's interactions with the environment. We offer two complementary implementations for this step. The second step analyzes the trajectories of these features, and computes a clustering of features and their trajectories into rigid bodies. The third component of our algorithm determines the kinematic constraints between pairs of rigid bodies.

4 Collecting Perceptual Information

The first step of our algorithm collects perceptual information. We rely on the robot's manipulation capabilities to create object motion. By physically causing objects to move, the robot generates a strong perceptual signal for object segmentation. Here, the robot's motion is scripted. However, this restriction can be removed by learning interaction strategies, as demonstrated by Katz et al. [7].

Once motion is induced, the robot collects perceptual information by tracking visual features using an RGB-D sensor. Thus, for every tracked feature, we have both color and depth information. Our perceptual skill uses one of two types of visual fea-

tures and corresponding tracking algorithms. The first is the Lucas-Kanade (LK) feature tracker, using Shi-Tomasi corner features [11]. The second is based on matching scale invariant SIFT features between key-frames [10].

The LK tracker assumes that brightness remains consistent between the same pixels from one frame to the next, and that only small movements occur between frames. The tracker estimates the velocity of a moving point feature by the ratio of the derivative of the intensity over time divided by the derivative of the intensity over space, and the resulting algorithm can track in real-time a large number of features. However, the LK feature tracker poorly handles occlusion: if the robot's arm blocks parts of the scene, all features that are temporarily invisible are lost or become unreliable.

SIFT features [10] overcome this limitation of the LK tracker. These features describe interesting points in an image so as to be robust to changes in scale, noise, and illumination enabling features to be matched between images despite significant changes in position. Using SIFT features provides robustness against occlusion: a feature that cannot be found in a certain frame due to occlusion is not declared lost; it may be detected in a later frame. Computing SIFT features descriptors and matching across frames is significantly slower compared to the LK tracker, so our implementation matches SIFT features in a sparse sequence of images containing 10% of the frames used by the LK tracker.

Throughout interaction with the environment our robot tracks a set of features. The robot records the features' images coordinates (u, v), 3D coordinates (x, y, z), and color values (r, g, b) for each time t: $f_i(t) = \{u, v, x, y, z, r, g, b\}$. This process of collecting perceptual information is identical, regardless of whether features are generated by the LK tracker or SIFT matching.

Feature tracking is a simple operation. It only requires that the scene contains sufficient texture to support visual tracking. It makes no assumption about the shape, size, or color of objects, about their motion, or the motion of the camera. However, both SIFT and LK feature tracking in unstructured scenes are highly unreliable. LK features can jump between image regions, are lost, swapped, or drift along edges in the image. And SIFT features can be wrongly paired. The next step of the algorithm will automatically eliminate this noisy data, rendering the algorithm suitable for manipulation in unstructured environments.

5 Segmenting Rigid Bodies

The second step of our algorithm computes rigid-body segmentation. To segment an unknown scene into rigid bodies, we leverage the fact that features associated with a single rigid body often share similar spatial, temporal, and appearance characteristics. Color and texture consistency over a spatially contiguous region may imply similarity of material. The 3D distance between features indicates spatial proximity, and 3D feature trajectories expose relative motion.

Although the 3D relative motion provides the strongest evidence to determine that two features belong to the same rigid body, other cues are important for two principled reasons. First, the relative motion may be small, as desired when manipulating an unknown object. In particular, the trajectories of features that are close to an axis of

Figure 3: CD case; visual features marked as yellow, blue, and red circles. The relative motion between the yellow and blue features is small. Additional cues such as texture consistency and spatial proximity are necessary to improve segmentation.

rotation will be minimal (see Fig. 3). Also, we believe that the additional cues are necessary for learning manipulation strategies from experience, ultimately enabling segmentation without interaction [7].

All of these clues exploit the structure of the problem but by themselves are insufficient to compute object segmentation. Our algorithm therefore integrates all these clues to generate a combined object segmentation hypothesis. Segmentation hypotheses are captured in a fully connected multi-graph $G = (V, E)$. A vertex $v \in V$ corresponds to either an LK or a SIFT feature f_i in the image and contains the feature observations $f_i(t) = \{u, v, x, y, z, r, g, b\}$. The weight $w(e_{i,j})$ of an edge $e_{i,j} \in E$ is a probability, indicating the belief that $f_i(t)$ and $f_j(t)$ belong to the same rigid body. We define three predictors, each considering the above clues. These predictors determine the probability that two features belong to the same rigid body. We assume that the predictors are independent and combine their outcome using the Naive Bayes rule: $w(e_{i,j}) = \prod_k P_k(f_i, f_j)$, where $P_k(f_i, f_j)$ is the probability computed by predictor k. This graph representation is similar to [8], however, our vertices are augmented with 3D information. As a result, our algorithm relies on a smaller, more efficient and reliable, set of predictors. In contrast with [8], our predictors have access to 3D measurements, and therefore have a physical meaning (3D distance and motion), resulting in improved segmentation performance (Fig. 5).

Relative 3D Motion Predictor: The distance between two features f_i and f_j that belong to the same rigid body should remain approximately constant over time. The relative motion predictor leverages this insight by computing δ, the maximum change in distance between f_i and f_j over time. If δ is below a noise threshold of ϵ_1, we conclude that f_i and f_j are likely to belong to the same rigid body. The probability that f_i and f_j are connected decreases linearly, until $\delta = \epsilon_2$, where we consider the features disconnected.

$$w_{i,j} = \begin{cases} 1 & \text{if } \delta(f_i(t), f_j(t)) \leq \epsilon_1 \\ \frac{\epsilon_2 - \delta}{\epsilon_2 - \epsilon_1} & \text{if } \epsilon_1 \leq \delta(f_i(t), f_j(t)) \leq \epsilon_2 \\ 0 & \text{otherwise} \end{cases}$$

3D Distance Predictor: If the distance between two features f_i and f_j is small, they are more likely to belong to the same rigid body than to different bodies. The 3D distance predictor leverages this heuristic. It computes a confidence value as a function

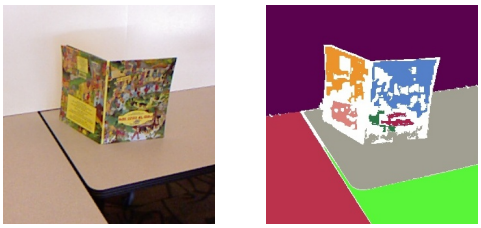


Figure 4: Illustration of the Color and Texture predictor.

of the distance $\delta(f_i, f_j)$ between the features before and after the interaction.

$$w_{i,j} = \begin{cases} 1 & \text{if } \delta(f_i(t), f_j(t)) \leq \epsilon_3 \\ \frac{1}{2} + \frac{\epsilon_4 - \delta(f_i(t), f_j(t))}{2(\epsilon_4 - \epsilon_3)} & \text{if } \epsilon_3 \leq \delta(f_i(t), f_j(t)) \leq \epsilon_4 \\ \frac{1}{2} & \text{otherwise} \end{cases}$$

Color and Texture Predictor: The color and texture predictor exploits the fact that image regions sharing similar color and texture are more likely to be part of the same rigid body. It uses the mean-shift color segmentation algorithm to segment an image into color-consistent regions (see Fig. 4). It assumes that point features that are in the same color region are more likely to be on the same rigid body. The predictor is neutral towards features that are associated with different color and texture regions.

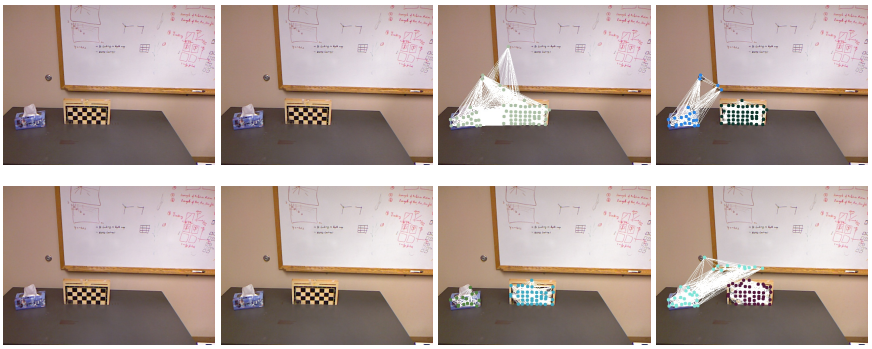
We note that the parameters $\epsilon_1 = 0.5cm$, $\epsilon_2 = 0.75cm$, $\epsilon_3 = 2cm$ and $\epsilon_4 = 3cm$ were chosen based on the precision of the RGB-D sensor. The probabilities set by each predictor were chosen based on experience. In future work, we intend to learn these parameters from labeled segmentations [15].

To extract a rigid-body segmentation hypothesis from the resulting graph, we first discard edges with weight zero. We rely on an efficient implementation of weighted max-flow as described in [3] to recursively decompose the graph into strongly connected components. The recursion terminates when decomposing a graph requires removing more than half of its edges. Each of these components represents a rigid body. Our algorithm therefore integrates clues about spatial proximity, color and texture similarity, and relative motion to generate a rigid-body segmentation.

6 Detecting Kinematic Constraints

The last step of the algorithm determines the kinematic structure of an object. The input to this step is a set of rigid bodies, each represented by a cluster of point features and their trajectories. Our goal is to determine, for every pair of rigid bodies, whether they are connected by a revolute joint, a prismatic joint, or are disconnected.

The kinematic relationship between a pair of rigid bodies is determined by their relative motion. Fig. 6(a) and 6(b) show the trajectories of two sets of point features, each associated with one side of a checkerboard. Because the two bodies move together, while also rotating with respect to each other, it is difficult to determine that they are



(a) Initial pose (b) Translation of 1cm (top) and 4cm (bottom) (c) [8]'s Segmentation (d) Our Segmentation

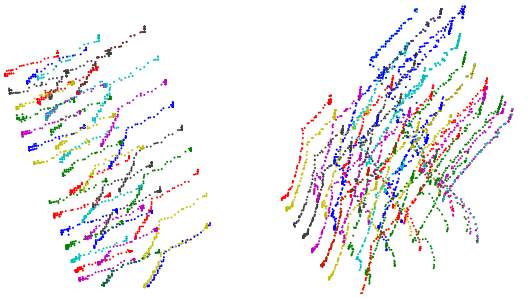
Figure 5: Comparing our segmentation performance and [8]. Our method requires smaller displacements, and therefore separates the board from the background after a translation of only 1cm, compared to 4cm for [8].

connected by a revolute joint. Removing the common global motion, however, provides very clear evidence for the revolute joint (Fig. 6(c)).

To extract the relative motion between pairs of rigid bodies, we first compute the relative homogeneous transformation ${}^0H_k$ for one rigid body between its initial pose at time t_0 and its pose at every time t_k along its trajectory. To handle measurement noise, we find the optimal transformation by solving a least square minimization problem using singular value decomposition (SVD) [2]. We obtain the relative motion trajectories between the bodies by applying the transformations ${}^0H_k$ computed for the first body to the feature trajectories of the second body (Fig. 6(c)).

The relative motion trajectories between two rigid bodies reveal their kinematic relationship. If the bodies are connected by a prismatic joint, the relative motion will have the form of straight, parallel lines of equal lengths. Our algorithm calculates the lengths of all relative motion trajectories. If the lengths are approximately equal, we use RANSAC line fitting, and calculate a fitness measure for each relative motion trajectory. A good fit indicates a **prismatic joint** (Fig. 7).

If the prismatic fitness measure is low, we check whether the relative motion can be explained by a **revolute joint**. Again, we use RANSAC, but this time to fit a circle to each trajectory. We then fit a line through the centroids of these circles. If the line is close to the majority of the centroids, the relative motion is due to a revolute joint, and the line is its axis of rotation (Fig. 6(c)). Otherwise, we declare the two bodies to be **disconnected**. Fig. 6(d) shows a complex motion between one side of the checkerboard and the static background. As expected, the trajectories cannot be explained by a single degree-of-freedom joint.



(a) Left board motion

(b) Right board motion

(c) Relative motion between the boards: revolute joint detected

(d) Relative motion board-background: declared disconnected

Figure 6: Feature trajectories for a checkerboard. The two sides of the board move with respect to the background, and rotate with respect to each other.

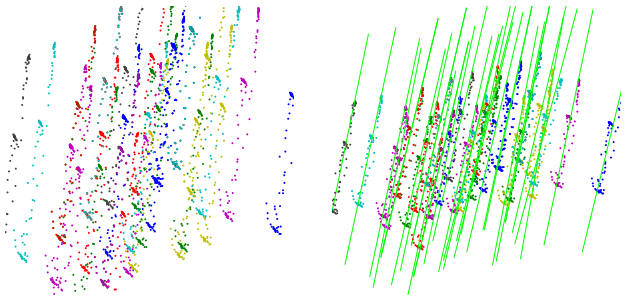


Figure 7: Left: motion trajectories of a drawer. Right: successful RANSAC line fitting to the relative motion.

7 Experimental Validation

We validate the proposed perceptual skill for segmenting, tracking, and modeling 3D rigid articulated objects in 9 real-world experiments. Our goal is to demonstrate that the robot can segment the target object and its rigid parts, track the position of each rigid body over time, and identify the kinematic relationship between rigid bodies. We

will also show that our method applies whether interactions are created by the robot or by a human teacher.

We conducted two types of experiments. In the first, our mobile manipulation platform (Fig. 2) interacted with 6 different real-world objects (see Fig. 8(a)). And in the second, the robot observed, while a person interacted with 3 additional objects (see Fig. 8(b)). The objects used in both types of experiments vary in scale, shape, color, texture, and kinematic structure. A cheap off-the-shelf RGB-D sensor, the Microsoft Kinect, provides the robot with 15 frames-per-second of color and depth information with a resolution of 640×480 pixels. This sensor co-registers the color image with a depth map.

Fig. 8(a) shows 6 experiments in which interaction was generated by the robot. The figure illustrates the performance of our algorithm in segmenting rigid bodies in an unstructured scene and modeling the kinematic constraints between bodies. Each row shows one object: a checkerboard, a stroller, a train, a book, a drawer, and a folder. The first column shows the object before the interaction, the second shows the object after the interaction, the third shows the clustering of the tracked features into rigid bodies (edges of the graph are shown in white, clusters are color coded). And the last column illustrates the joints detected between the rigid bodies (green lines). In the first experiment, the robot interacts with a checkerboard by creating a motion between the board and the background, as well as between the two parts of the board. The segmentation into rigid bodies is accurate. Joint detection assigns high confidence values to a disconnected joint between the background and the board. It also discovers the revolute joint between the two parts of the board. Our algorithm works in this case because it can detect and analyze any number of rigid bodies, even when moving simultaneously. The second experiment shows an interaction with a baby stroller. The robot pushes the stroller along a straight line, creating a translation between the stroller and the background. . The algorithm correctly segments the stroller from the background and discovers a prismatic joint. More interaction with the stroller would reveal that it is disconnected from the background. We note that the sensor's resolution does not allow feature tracking on the wheels, and therefore the wheels' revolute joints are not discovered. In the third experiment, the robot slides a Lego train along its tracks. The train can only translate in its track, and therefore the robot segment the train from the background, and correctly detects a prismatic joint. The fourth experiment shows an interaction with a book. The robot only pushes one side of the book. Therefore, the other side of the book is associated with the background (no relative motion), while a revolute joint is detected between the moving side of the book and the background. We note the slight offset in the joint's position. This is due to some flexibility in the spine of the book. Nevertheless, the acquired model is good enough for manipulating the book. In the fifth experiment, the robot pushes a drawer. The algorithm segments and tracks the drawer with respect to its background. It correctly identifies the drawer's prismatic joint. The sixth experiment shows an interaction with a document folder. The robot pushes the folder, and interacts with the top part. As a result, three rigid bodies are detected: the two parts of the folder, and the static background. A revolute joint is identified between the parts of the folder. Due to flexibility at the joint, the observed motion is only approximately a rotation. Thus, the position and orientation of the joint are significantly offset. Future work should consider other types of joints. This exper-

iment too demonstrates that our algorithm is able to detect and analyze any number of rigid bodies, even when moving simultaneously.

Fig. 8(b) shows 3 experiments in which interaction was generated by a human teacher. The figure illustrates the performance of our algorithm in segmenting the various rigid bodies and modeling the kinematic relationships. Each row shows one object: a roll of paper-towel, a window, and an entire kitchen cabinet. Figures are organized in columns showing: the object before the interaction, the object after the interaction, the detected rigid bodies, and the detected joints.

The first experiment shows an interaction with a roll of paper-towel. Here, the human teacher rotated the roll around a central metal axis. The algorithm correctly segments the scene into two rigid bodies: the roll of paper-towel and the static background. It correctly identifies the revolute axis. In the second experiment, the human demonstrator closes a sliding window. The algorithm correctly segments the scene into two rigid bodies: background and moving window. It also detects the right kinematic relationship: a prismatic joint. Our third experiment shows a complex interaction with a kitchen cabinet. The demonstrator operates two drawers and a door. The different motions overlap. Our perceptual skill correctly segments the scene into four rigid bodies: background, top drawer, bottom drawer, and door. It identifies the prismatic joints connecting the two drawers to the background, and a revolute joint between the door and the background. It also determines that the drawers and the door are disconnected. We note that the smooth motion of the door and the lack of flexibility in the mechanism enables a very accurate detection of both position and orientation of all joints.

In all experiments, the proposed algorithm tracks the position of all features, and therefore the position of each rigid body. In addition, the algorithm tracks the configuration of the object. For a revolute joint, the configuration is the angle between the relevant rigid bodies. For a prismatic joint, it is the translational displacement between the bodies. Fig. 11 demonstrates the performance of our algorithm in tracking the configuration of the revolute joint of the checker board.

In each experiment, the proposed algorithm detected, segmented, and tracked all rigid bodies containing a sufficient number of visual features. The algorithm successfully obtained the kinematic structure in 9 out of 9 experiments. It detected the position and orientation of the joint correctly in 8 cases. In the folder experiment, the detected joint is offset due to inherent flexibility near the joint. Experiments were performed under uncontrolled lighting conditions, different sensor positions and orientations, and for arbitrary initial and final poses of the objects (see Fig. 9 and 10). The demonstrated robustness, effectiveness and repeatability provide evidence that this perceptual skill is suitable for manipulation in unstructured environments. The skill also transparently allows for learning from demonstration, an important feature for manipulation in human environments. We do not have ground truth information for the kinematic models of the above objects and therefore rely on visual inspection to judge the effectiveness of our method. Ultimately, we will combine the proposed perceptual skill with manipulation skills, which will allow us to determine whether the accuracy of our skill is sufficient to enable autonomous manipulation of unknown objects. However, given the above results we are confident that this is the case.

The greatest limitation of the our algorithm is its dependency on the presence of trackable visual features. A higher-resolution sensor would reduce this dependency.

We are also actively working towards the development of visual features that do not dependent so strongly on texture.

The runtime of all three steps of the algorithm depends on the number of tracked features as well as the number of rigid bodies in the scene. In our experiments, scenes were composed of 2-4 rigid bodies. When using LK features, the algorithm initially attempts to locate 500 features. In most cases, about half of the features are lost during tracking or discarded because depth information is missing. When using SIFT features, the algorithm is usually able to detect about 100 features in multiple frames. The runtime of the algorithm averaged 20 seconds, a 15-30 times improvement compared to the state-of-the-art approach [8].

8 Conclusion

We presented a perceptual skill for manipulation in unstructured environments. This skill enables autonomous segmentation, tracking, and modeling of 3D articulated objects. This ability is a prerequisite for purposeful manipulation. It enables a robot to monitor the progress of a manipulation task, detect its completion and identify failures. To achieve this, we rely on interaction with the environment to excite the explored degrees of freedom. This interaction can be performed by the robot or by human teacher. The proposed skill analyzes the outcome of the interaction to determines the shape, configuration, and kinematic model of the observed objects.

Our experiments showed the successful acquisition of 3D kinematic models of 9 real-world objects. The robot requires no prior knowledge of the objects. The resulting kinematic models were accurate, even in the presence of substantial noise in tracking due to the low resolution of the Kinect. The algorithm is also robust against small object deformations and joint flexibility. The skill only depends on a sufficient number of trackable features on each of the rigid objects in the scene, and on the ability to generate motion.

First, its run time is only a few seconds. Second, it allows for occlusions during the interaction. Third, it requires very small object motions—an important property when interacting with unknown objects. And most importantly, it makes no assumptions about the shape and kinematic structure of objects.

Our algorithm has four important advantages compared to the state of the art: its run time is over an order of magnitude faster, it can handle occlusions during the interaction, it requires very small object motions, and it makes no assumptions about the shape and kinematic structure of objects.

References

[1] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active Vision. *IJCV*, 1 (4):333–356, January 1988.

[2] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-Squares Fitting of two 3D Point Sets. *IPAMI*, 9(5):698–700, September 1987.

[3] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IPAMI*, 26:1124–1137, 2004.

[4] Paul Fitzpatrick. First Contact: An Active Vision Approach to Segmentation. In *IROS*, pages 2161–2166, 2003.

[5] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002. ISBN 0130851981.

[6] Dov Katz and Oliver Brock. Manipulating Articulated Objects with Interactive Perception. In *ICRA*, pages 272–277, Pasadena, CA, USA, May 2008. IEEE Press.

[7] Dov Katz, Yuri Pyuro, and Oliver Brock. Learning to Manipulate Articulated Objects in Unstructured Environments Using a Grounded Relational Representation. In *RSS*, pages 254–261, Zurich, Switzerland, June 2008.

[8] Dov Katz, Andreas Orthey, and Oliver Brock. Interactive perception of articulated objects. In *ISER*, India, 2010.

[9] Jacqueline Kenney, Thomas Buckley, and Oliver Brock. Interactive Segmentation for Manipulation in Unstructured Environments. In *ICRA*, pages 1343–1348, Kobe, Japan, May 2009. IEEE Press.

[10] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110, 2004.

[11] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *IJCAI*, pages 674–679, Canada, August 1981.

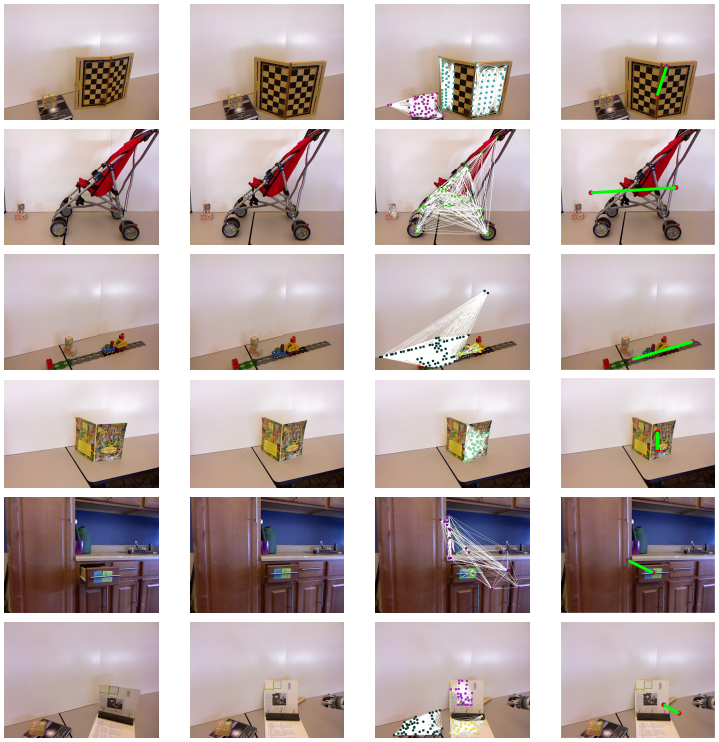
[12] David A. Ross, Daniel Tarlow, and Richard S. Zemel. Unsupervised Learning of Skeletons from Motion. In *ECCV*, pages 560–573, Germany, 2008. Springer-Verlag.

[13] Jurgen Sturm, Advait Jain, Cyrill Stachniss, Charlie Kemp, and Wolfram Burgard. Operating Articulated Objects Based on Experience. In *IROS*, Taiwan, 2010.

[14] Jurgen Sturm, Kurt Konolige, C. Stachniss, and W. Burgard. Vision-Based Detection for Learning Articulation Models of Cabinet Doors and Drawers in Household Environments. In *ICRA*, Alaska, May 2010. IEEE.

[15] Srinivas Turaga, Kevin Briggman, Moritz Helmstaedter, Winfried Denk, and Sebastian Seung. Maximin affinity learning of image segmentation. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *NIPS*, pages 1865–1873, 2009.

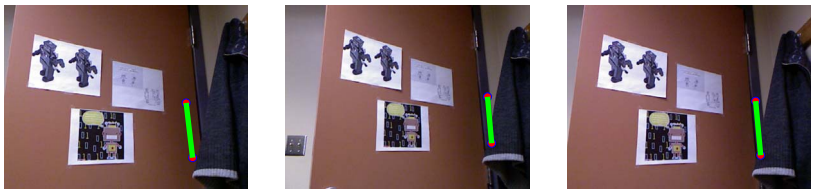
[16] Jingyu Yan and Marc Pollefeys. Automatic Kinematic Chain Building from Feature Trajectories of Articulated Objects. In *CVPR*, pages 712–719, USA, 2006.



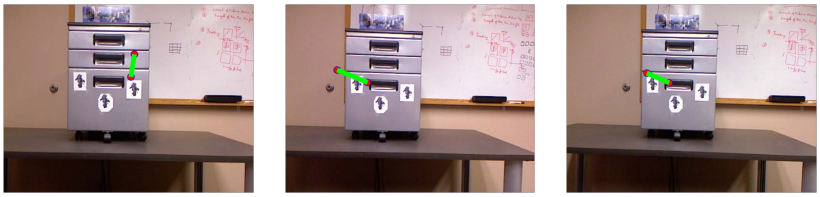
(a) Interactions generated by the robot

(b) Interactions generated by a human teacher

Figure 8: Experimental results showing the process of segmenting rigid bodies and detecting degrees of freedom in a scene. Interactions are generated by the robot in 8(a), and by a human teacher in 8(b). Left to Right: the object before the interaction; the object after the interaction; the results of segmenting the graph of tracked features into clusters of features on the same rigid body; the detected joints (marked in green). Videos are available at http://www.youtube.com/playlist?list=PLBB08C0290B79905C



(a) Detecting a door's hinge (10 trials, conditions vary)



(b) Detecting a drawer's prismatic joint (10 trials, conditions vary)

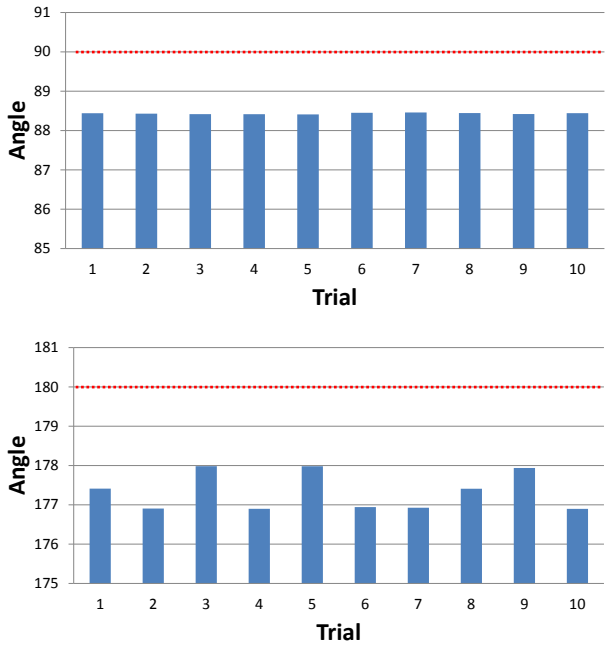


Figure 9: To demonstrate repeatability we conducted experiments with two objects: door and drawer. Each experiment was repeated 10 times, while changing lighting, sensor position and orientation, and initial and final object configuration. Three examples of the detected joint (marked in green) are shown for each object. The plots show the angle between the detected axis and the floor. Door: mean = $88.43°$, std. dev. = 0.016, Drawer: mean = $177.3°$, std.dev. 0.94. Results are close to expectations ($90°$ and $180°$). The difference ($\leq 3°$) is due to misalignment of the sensor with the floor.

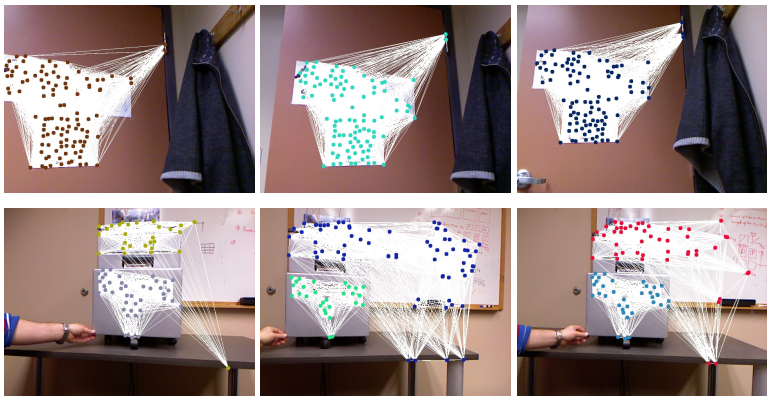


Figure 10: We performed 10 experiments with a door and 10 with a drawer under varying lighting conditions, sensor position and orientation, and object initial and final configuration. Segmentation results are consistent and repeatable.

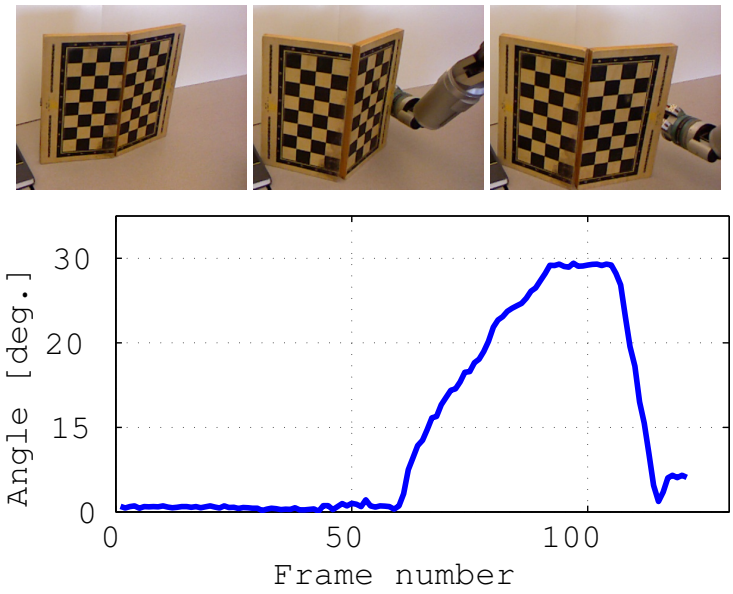


Figure 11: Tracking the configuration of a checkerboard: the robot translates the object, rotates the right-half, then rotates it back. The plot shows the resulting change in configuration.