Interactive Perception: Closing the Gap Between Action and Perception

Dov Katz and Oliver Brock
Robotics and Biology Laboratory
Department of Computer Science
University of Massachusetts Amherst

***Abstract*— We introduce Interactive Perception as a new perceptual paradigm for autonomous robotics in unstructured environments. Interactive perception augments the process of perception with physical interactions, thus integrating robotics and computer vision. By integrating interactions into the perceptual process, it is possible to manipulate the environment so as to uncover information relevant for the robust and reliable execution of a task. Examples of such interactions include the removal of obstructions or object repositioning to improve lighting conditions. More importantly, forceful interaction can uncover perceptual information that would otherwise be imperceivable. In this paper, we begin to explore the potential of the interactive perception paradigm. We present an interactive perceptual primitive that extracts kinematic models from objects in the environment. Many objects in everyday environments, such as doors, drawers, and hand tools, contain inherent kinematic degrees of freedom. Knowledge of these degrees of freedom is required to use the objects in their intended manner. We demonstrate how a robot is capable of extracting a kinematic model from a variety of tools, using very simple algorithms. We then show how the robot can use the resulting kinematic model to operate the tool. The simplicity of these algorithms and their effectiveness in our experiments indicate that Interactive Perception is a promising perceptual paradigm for autonomous robotics.**

I. Introduction

Roboticists are working towards the deployment of autonomous robots in unstructured and dynamic environments. Adequate autonomy and competency in such environments would open up a variety of important applications for robotics, ranging from planetary exploration to elder care and from the disposal of improvised explosive devices to flexible manufacturing and construction in collaboration with human experts. For these applications, it is not possible to provide detailed *a priori* models of the environment. The ability to efficiently acquire and iteratively improve such models from perception is thus an essential prerequisite for autonomous operation in unstructured environments.

Perceptual techniques, in particular in the domain of computer vision, have recently made significant progress. Novel fundamental vision primitives, such as Lowe features [18], provide novel tools for extracting distinctive invariant features from images which can be used to reliably match different views of an object. An increasingly powerful set of tools is being developed to address complex vision problems. Researchers have also made significant progress in specific applications, such as semantic image retrieval [24] and semantic video search [22]. However, few of these advances in the realm of perception have had significant impact in robotics.

Fig. 1. Objects that possess inherent degrees of freedom; these degrees of freedom cannot be extracted from visual information alone, they have to be discovered through physical interaction

Why have the advances in computer vision not had significant impact in robotics? We believe that there are two closely related reasons. First and foremost, after initial and foundational work at the intersection of computer vision and robotics [13], both fields have progressed mostly independently. As a result, roboticists currently do not exploit the full potential of state-of-the-art computer vision techniques.

But there is a second important reason for the lack of impact made by recent progress in computer vision. We believe that adequate perceptual capabilities have to be developed in the context of a specific robotic task. The perceptual information extracted from the sensor stream can then be tailored to the task to provide the appropriate feedback to ensure its robust task execution, in particular in the presence of significant uncertainty. In contrast, the majority of computer vision research is concerned with general perception skills. The lack of impact such skills have had in robotics is a result of the difficulty in developing general, task-relevant perception skills.

The importance of considering the perception problem in the context of a specific task has been demonstrated in a highly visible manner by Stanford's robot Stanley during the 2005 DARPA Grand Challenge race. The vision techniques that helped Stanley win the race were effective because they were tailored to a specific problem [7].

In this paper, we introduce the concept of Interactive Perception. Interactive Perception exploits forceful interac-

tions with the environment to uncover adequate perceptual information for the robust execution of specific tasks. The restriction to a specific task facilitates the perception task, as only task-relevant information has to be extracted from the sensor stream. The inclusion of forceful interactions into the perceptual process makes it possible to extract information from the environment that would otherwise be unobtainable or could only be obtained with significant domain knowledge.

To illustrate the promise of Interactive Perception as a perceptual paradigm for autonomous robotics, we present early efforts towards the development of perceptual skills that extract kinematic models of the environment. Many objects in everyday environments possess inherent degrees of freedom that have to be actuated to perform their function. Such objects include door handles, doors, drawers, and a large number of tools, such as scissors and pliers.

Knowledge of their kinematic models is necessary for the successful execution of various tasks. Since it is impossible to provide an autonomous robot with a kinematic model for all objects in the environment, the robot has to be able to extract such a model from its surroundings.

In this paper, we show preliminary work towards interactive perception primitives that extract kinematic models from the environment. In our experiments, a robot interacts with a set of tools; the resulting sensor stream provides sufficient information to extract a model of their kinematics. This model is then employed to compute an action that transforms the kinematic state of the tool into a desired goal state, mimicking the use of the tool to achieve a task. We believe that the relative ease with which we are able to address this task makes a convincing case for the use of interactive perception as a perceptual paradigm for autonomous robotics.

II. Related Work

Successful manipulation depends on the sensory stream that is used to assess the state of the world. The classical approach (see [10]) towards streaming information from sensors to actuators analyzes a given stream of information in order to make a decision regarding the best course of action.

The problem of interpreting a stream of sensory information dates back to the first days of AI. Computer vision researchers explored extensively the problems of object segmentation and labeling from static images. These problems, which seems to be solved effortlessly by humans, were found out to be quite challenging. Researchers have realized that information about dynamic scenes that is acquired continuously over time is easier to understand, and that additional view points can provide important information.

Active vision is a strategy which makes the observer an active participant in the process of data acquisition [1], [**?**], [2]. In the vision domain, tremendous progress was made using this strategy [10]. Extracting structure from visual input, for example, was found to be much simpler when the camera's motion can be controlled by the observer [20], [27]. Moreover, by generating a camera motion, an important tool for many vision algorithms such as feature tracking, can be applied to a static object. Consequently, difficult problems such as depth estimation become trivial by the deliberate generation of optical flow [17].

Active vision represented a paradigm shift in which the agent is no longer a passive observer. The agent can actively affect the visual stream by controlling the position and orientation of its sensors. Along with it, the development of new tools and new applications was imminent. One such application is visual servoing [14], which can for example leverage visual sensory input into accurate position control of a robotic manipulator. In this example, the observer controls both the robotic arm and the camera, thus can actively change the visual stream. This scenario demonstrates how position control, one of the fundamental primitives of manipulation, can be greatly improved by interacting with visual input.

Controlling the movement of a camera reveals new information. However, in some cases, this process does not generate the data required to support a specific task. For example, object segmentation and predicting possible movement of rigid bodies in a plane remain great challenges even when camera movements can be controlled. Physical interaction with the world can remedy many of these difficulties. For example, object segmentation has been shown to be relatively simple when vision and manipulation interact. Instead of attempting to interpret a scene using cues from a static image, Fitzpatrick and Metta (see [9], [21]) actively poked objects using a robotic manipulator. Optical flow methods allowed them to identify which objects in the image had moved as a result of the forceful interaction with the manipulator, thus identifying which rigid bodies are not attached to each other. The objects' outlines were also retrieved. We see this work as the precursor of the Interactive Perception framework.

The task of predicting the movement of novel objects in the plane can be simplified by interacting with the objects. Christiansen et al. address this problem by placing objects on a tray which could be tilted by a robotic agent (see [3]). They show that by actively tilting the tray, a robot can increase its knowledge about the object's behavior. The robot learns about the object's response to the tilting actions and builds a model which allows the planning and execution of a desired object displacement. Thus, the complex task of understanding how an object can move is made simple by actively moving the observed object.

Stoytchev et al. have used a predefined set of interactions with a rigid object (tool) to explore the tool affordances [25]. They extracted the movement of rigid bodies in response to intentional poking by the robot. The acquired knowledge can later be applied during a task execution stage. The complex task of predicting the behavior of rigid bodies becomes much easier when the robotic agent is allowed to deliberately affect its environment.

The last three examples demonstrate the positive effects that deliberate action has on the successful completion of tasks and on the difficulty of the perception problem. The interaction between the agent and its environment reveals sensory information relevant to the task at hand.

Active vision has turned sensing from a passive to an active process of data collection. The next natural step seems to be the ability to actively change the world to further increase sensor range. The following section discusses this new paradigm, and explains how it can dramatically improve the capabilities of a robotic agent in an unstructured and dynamic environment.

III. Interactive Perception

Robots live in the physical world and as such have to face the challenges of this world. Being an embodied agent, however, also holds some promise: robots are not restricted to be passive observers. Robots can direct their interactions with the world in specific ways to facilitate the execution of a task.

Developing robots that can operate in a dynamic environment is the major challenge for roboticists. One important aspect of this problem is perception. The difficulty arises from the amount of uncertainty inherent to the sensing process: two sensor readings of the same object can be quite different due to lighting conditions, object composition, and obstructions. Actively directing perception by deliberate interaction with the environment can provide a remedy to this difficulty.

Many perceptual tasks can be greatly facilitated by physical interactions with the environment. Such interactions can remove obstructions, provide an easy and controlled way of exposing multiple views of an object, or can alleviate the negative effects of lighting conditions by moving objects in the field of view. Other perceptual tasks are difficult or even impossible to accomplish without interacting with the environment. For example, reading the text in a closed book, checking whether a door is locked, and finding out what is the purpose of a switch mounted on the wall. Physical interactions thus can make traditional perceptual tasks easier and they make a new class of perceptual information accessible to a robotic agent.

The promise of Interactive Perception is supported by examples from the development of physical and mental skills in humans. During the acquisition of physical skills by infants, for example, physical interactions with the environment are necessary to bootstrap the cognitive process of learning the connection between action and effect, the kinematics of the own body, and the properties and functions of objects in the environment.

The related work section has outlined the progression of computer vision towards active vision. We presented several examples of perceptual processes that were aided by physical interactions of a robotic agent. We refer to this new approach as Interactive Perception. Interactive Perception continues the progress from static image analysis to active vision by integrating action and perception into a single, synergetic process.

However, the capability to interact with the environment as part of the perceptual process also incurs the additional cost of choosing and executing the most adequate interaction for a specific perceptual task. Because a large number of possible interactions may be available to the robotic agent, this is a challenge. Clearly, the agent should choose the action that promises maximum progress towards accomplishment of the task. For Interactive Perception to be fully effective, perceptual primitives have to be integrated into a framework that allows the selection of the most adequate perceptual interaction.

Active learning is a branch of machine learning that attempts to dynamically focus the learning process of a learning agent (see [4], [5], [11]). The learning agent can select its training data incrementally and in response to all the information acquired previously. In machine learning tasks, this active learning process has been shown to be highly effective.

Interactive Perception has to be integrated into an active learning framework to allow the robotic agent to incrementally extract task-relevant perceptual information from the environment until the task can be accomplished successfully. This integration of active learning and Interactive Perception will be the subject of future investigations. In this paper, we focus on a specific perceptual primitive to demonstrate the effectiveness of interactive perception.

One important category of perceptual information that can be accessed through physical interactions is the inherent kinematics of articulated objects in the environment. A kinematic model enables a robotic agent to predict the behavior of objects. This ability is essential in the context of tool use, for example. Real world environments are abundant with articulated bodies: doors, door handles, drawers, light switches, and hand tools, to name only a few. To use these objects in a purposeful manner, i.e. to open a door or a drawer, or to use a tool, a robotic agent has to understand the kinematics of these objects. Therefore, understanding the kinematics of articulated bodies is a crucial first step in the process of developing robotic agents that can operate in unstructured and dynamic environments.

The following sections of this paper demonstrate the effectiveness of Interactive Perception in the context of a specific perceptual task—a task that is impossible to accomplish with a close integration of action and perception. The task is intended to capture the complexities of tool use for tools with inherent degrees of freedom that have to be actuated to perform the tool's function. Our robot is presented with a set of tools and interactively builds a kinematic model. The kinematic model is required to predict future interactions with the tool, and more specifically to learn the functions that can be performed using that tool. We will present a simple Interactive Perception algorithm which uses a manipulator and a video camera to learn how tools such as scissors, shears, players or staplers can be used. The acquired kinematic model is then used by the robot to predict an interaction with the tool that accomplishes a given task, mimicking the use of the tool.

IV. Obtaining Kinematic Models Through Forceful Interactions

We now describe how Interactive Perception can be used to the extract the kinematic properties of an object. We present algorithms to construct a kinematic model for previously unseen objects. We will show how by interacting with unknown objects, a robot is easily able to recover their kinematic properties. The robot can subsequently use this model to predict the appropriate interaction for tool use.

Since our primary goal is to show the promise of interactive perception as a perceptual paradigm, several simplifying assumptions were made. To facilitate the computer vision aspects of our task, we have placed all objects on a plain white background, and randomly glued stickers to add attractive features. Moreover, we have assumed that the motion takes place in a plane orthogonal to the image plane. In order to simplify the manipulation task, we manually programmed a trajectory for the manipulator. Finally, we assumed objects with exactly one degree of freedom (one joint). In our future work, we plan to successively remove all of these restrictions.

The vision-related simplifications are relatively easy to remove. A big body of research exists on tracking in cluttered environments without special markers. Removing the 2D assumption could be done using active vision techniques. The robot controls the position and orientation of the camera, and therefore can align it with the plane of motion. Moreover, proprioception can be used to bootstrap the alignment process. The robot can collect spatial information regarding the exact position of an object. This data enables accurate depth perception, thus allowing the robot to improve the positioning of the camera. Not surprisingly, removing the vision-related assumptions requires further integration between vision and manipulation.

Since the robot also executes the motion, this is relatively simple. In order to interact with the object without a preprogrammed trajectory, we plan to start our exploration process with active segmentation similarly to [9] and [21]. This includes an initial random phase were the manipulator swaps the environment in an attempt to segment objects. The interaction may provide interesting objects, for which we might want to construct a kinematic model. Finally, extending the work to multiple joints is a trivial extension, as will be indicated in the next subsection.

A. Algorithm

In this section we describe a simple algorithm which allows, by means of interaction with the environment, the construction of a kinematic model. Our algorithm builds a DH parameter description [6] of a given kinematic chain, and uses the obtained model to create a plan for forming a right angle between the two links of the chain.

Any kinematic model requires the identification of the joints in the kinematic chain. The key insight behind the algorithm is that the relative distance between two features on a rigid body does not change as the body is being manipulated. Moreover, the distance between points on two different links connected by a joint does change as the links rotate about the joint. Using these two observation, we can conclude that the distance between any point on any one of the two links and a point on the joint does not change.

Tracking a set of features of an object allows us to compute the distance between any two features. One would expect to be able to distinguish three groups of features: features on link 1, features on link 2, and features on the axis. Features in one of the first two groups will not move with respect to each other, but may move significantly with respect to features in the other group. The third group is a set of features which lie on the joint, or on the two links simultaneously. Therefore, those features belong to both the first and the second group. While our manipulator interacts with the object, our vision system tracks features on the object. We separate the features into the three groups, and consequently find the features which represent the axis.

Formally, let us define a graph $G = (V, E)$ where V represents the set of tracked features. An edge $e = (v_i, v_j) \in E$ exists in the graph if and only if the distance between v_i and v_j does not change during the interaction (after accounting for measurement error). The resulting graph contains vertices with high degrees, which represent features that lie on a joint (connected to features on both links). Removing features with high degrees breaks the graph into two separated components; each one represents the set of features on one of the two links.

In order to describe the links of the object, we can construct a convex hull around the points of each group. Tracking enough features increases the match between the convex hull and the actual shape of the link. The length of each link is taken to be the distance between the furthest point in each group and the joint. We use this newly acquired knowledge to instantiate a DH-parameter model for planar kinematic chains. This model is used later to predict the necessary action in order to manipulate the kinematic chain into a cross shape.

The following subsections describe in details the implementation of the kinematic model building algorithm. It is worth noting that the specific way in which we choose features, track them or analyze their relative motion does not affect the algorithm.

B. Tracking objects

Our vision system uses the open source computer vision library OpenCV [15]. OpenCV provides us with an easy interface to the camera (image capture and recording), and contains a very rich set of image processing tools. We track a set of image features, and store the results in a matrix form. For each feature, we record its position in each frame.

We use OpenCV's API for feature selection and tracking. Feature selection is accomplished using OpenCV's API (cvGoodFeaturesToTrack). In principle, features are selected by finding corners with big eigenvalues in the image. We then compute the eigenvalue for every source image pixel, perform non-maxima suppression (leaving only local maxima in a 3x3 neighborhood), and reject corners with eigenvalues below a certain quality level threshold. Finally, we keep a

predefined distance between features. If a couple of features are clustered together around an attractor, we gradually remove features (starting with the weakest feature) until the desired distance between features is achieved.

Once features are selected, we need to track their change in position between frames. To that end we use an optical flow algorithm. OpenCV's implementation of optical flow is based on Lucas and Kanade's algorithm (see [19]).

C. Constructing a graphical representation

Every kinematic chain is composed of links and joints. Therefore, the first task we perform is axis detection. Let us consider the tracked features in the case of two links connected by a joint. Two features on different links (different rigid bodies) have a non constant distance as the links move independently about the joint. In contrast, the distance between two features that are on the same link remains constant.

We use these trivial observations to build a graph based on the maximal change in distance between two features. Every node $v \in V$ in the graph represents a tracked feature in the image. An edge $e \in E$ connect nodes (v_i, v_j) if and only if the distance between v_i and v_j remains below a threshold. This threshold represents our tolerance for noise. We assume that changes in distance below the threshold represent points that keep constant distance with respect to each other. Since we get many measurements for the distance between two points (one for each frame), we define the distance as the maximal distance over all frames.

The resulting graph will be used by the algorithm described in the following section. This algorithm works only for rotational joins. Kinematic chains may have other types of joints (i.e. prismatic, spheric, etc.). We plan to remove this limitation in future work.

D. Graph separation and model building

In the previous section we described the construction of a graph $G(V, E)$ in which the nodes V represent the tracked features in the image, and the edges E connects two nodes that are on the same rigid body. We now process this graph to learn the position of the axis, and to separate features into two groups (one group for each rigid body).

First, we begin with finding the axis in an object using the graph. As explained in the previous section, nodes on the same rigid body are likely to be connected to other nodes on the same rigid body. A node that represents a feature on an axis is connected to nodes on both rigid bodies that are connected through the axis. Consequently, nodes on an axis are nodes with the highest degree in the graph. Finding features that are clustered around the axis therefore merely requires to find the center of mass of a few highest degree nodes.

Second, we address the problem of separating the graph into two groups of nodes, where each group represents a different rigid body. We note that an edge between nodes means that they are likely to be on the same rigid body. In order to find the number of separated components in the graph we proceed as follows:

1) Pick a random node and assign to it a color
2) Recursively color all its neighbors until all nodes in one connected components have the same color
3) Proceed to the next component, until all nodes are colored

The number of colors used tells us the number of separated components in the graph. In the beginning, we are likely to find one component in the graph as nodes that represent features around the joint are connected to nodes that represent features on both links. We gradually remove nodes with high degrees, until the two components are separated.

When the iterative process is over, we are left with two groups of nodes which reliably represent two different rigid bodies. The next step is getting a crude description of the outline of the object in hand; we rely on OpenCV's implementation of Sklansky's algorithm for finding the convex hull of a group of points. The details of the algorithm are not important here. Instead, it is worth noting that the result is the minimal polygon which contains all the points in the set. For us, the result means a rough outline of the two rigid bodies in the image. The exact shape of the outline depends on the tracked features as well as the points that were removed in order to separate the graph. In the future, we plan to use active segmentation (i.e. [9], [21]) to extract a more accurate outline of the rigid bodies.

The task that the robot executes is the manipulation of each tool into a cross shape, that is forming a right angle between the tool's links. We choose two points, one on each link, which are the furthest from the joint. The distance between each point and the joint is taken as the respective link's length. Using the information about the position of the joint and the length of the links we are able to construct a DH parameter model. This model provides us with a simple method for computing the affects of the displacement of one link on the angle between the links.

The results of this section can be summarized as model building. We have shown how to find the axis and detect the rigid bodies. Consequently, we are able to construct a kinematic model of the given object, and plan future interactions with it. In the next section we describe experimental results of using this algorithm to learn the operation of 4 tools: scissors, shears, plier, and a stapler. Our robot builds a kinematic model for each tool, and then devises a plan for performing the task of setting a desired angle between the tool's links.

V. Experimental Results

The previous sections have outlined our approach towards extracting the kinematics of newly encountered objects. In this section we present our robotic platform *UMan* and discuss a set of experiments which illustrate the simplicity and effectiveness of Interactive Perception.

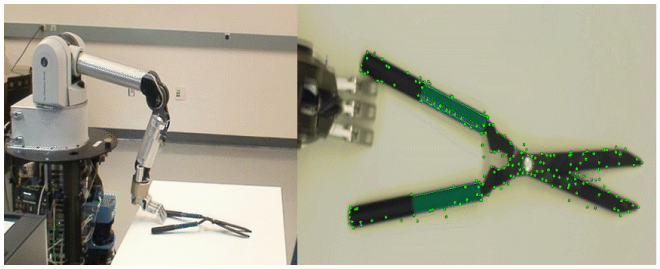

Fig. 2. UMan interacts with a tool by reaching its arm towards the tool. The right image shows the tools as seen by the robot, with dots marking the tracked features. The left image shows the experimental setting

A. System description

The experiments presented in this paper were performed using our mobile manipulator *UMan* (UMass Mobile MAnipulator). As Interactive Perception requires both sensing and manipulation, we will now describe the respective capabilities and limitations of our robotic platform. For a more detailed review of *UMan* see [16].

UMan consists of a holonomic mobile base with three degrees of freedom, a seven-degree-of-freedom manipulator arm, and a four-degree-of-freedom hand. The platform provides adequate end-effector capabilities for a wide range of dexterous manipulation tasks. We consider mobility as additional degrees of freedom in service to manipulation, rather than as an objective itself. Appropriately, *UMan*'s mobile base supports manipulation without imposing additional constraints.

Fig. 3. Barrett 7-DOF WAM

*UMan*s mobility is provided by a modified Nomadic XR4000 mobile base. Its four casters are dynamically decoupled [12] to provide holonomic motion, which facilitates a unified control scheme for degrees of freedom associated with mobility and manipulation. The XR4000 mobile platform was specifically designed for mobile manipulation. Its power system allows untethered operation for several hours. The base is sized to be able to contain adequate computational resources and sensors.

A Barrett Technologies Whole Arm Manipulator (WAM) [26] with seven anthropomorphic degrees of freedom (three in the shoulder, one in the elbow, three in the wrist, see Figure 3) together with the three-fingered Barrett hand provide *UMan*'s dexterous manipulation capabilities. All electronics for the control of the arm are built into the arm itself, facilitating its integration with a mobile platform. The WAM provides good dynamic performance and torque sensing in its joints. *Uman* is thus capable of using all of its links to perform force controlled manipulation tasks. The three-fingered Barrett hand (see Figure 4) can flex any of its three-link fingers individually. A fourth degree of freedom in the hand permits switching between an enveloping grasp to grasp with an opposing thumb.

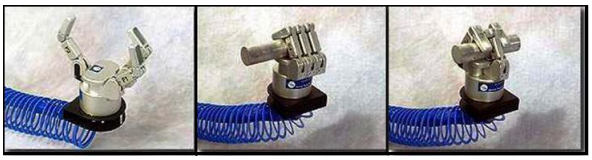

Fig. 4. Barrett 4-DOF Hand

UMan's mobile base houses two single-board PCs with Pentium 4, 2.4GHz CPUs. One of these PCs is dedicated to real-time control of the base and the manipulator arm. It runs the real-time operating system QNX [23]. The other PC runs Linux [8] and is dedicated to higher level tasks, such as vision processing, navigation, motion planning, and task planning. Both computers are connected via a dedicated wired Ethernet link.

UMan is equipped with a rich suite of sensors. A SICK LMS200 laser range finder enables navigation. Visual input is received from a Unibrain Fire-i web camera mounted on the wrist. The camera has an IEEE-1394 (Firewire) interface and can produce 30 frames per second at a resolution of 640 by 480 pixels. By controlling the position and orientation of its arm (and consequently of the camera), *UMan* surpasses the capabilities of many passive and active vision hardware systems. Linking vision and manipulation together allows *UMan* to look behind obstructions in the field of view and generate multiple views of the same object. Finally, *UMan* is equipped with two force sensors that improve its range of dexterous manipulation. The first is mounted on the fingertips of *UMan*'s Barrett hand (3 ATI Nano17 6-axis force/torque sensors). The second is an ATI Gamma 6-axis force/torque sensor, mounted on the wrist.

B. Experiments

We used four tools during the experimental phase: scissors, shears, plier, and a stapler. Each one of the tools has a single degree of freedom (one revolute joint). The only exception is the pliers which also have a prismatic joint. For the purpose of this paper we have ignored this prismatic joint.

The tools are off-the-shelf products and have not been modified for our experiments. They vary in scale and shape.

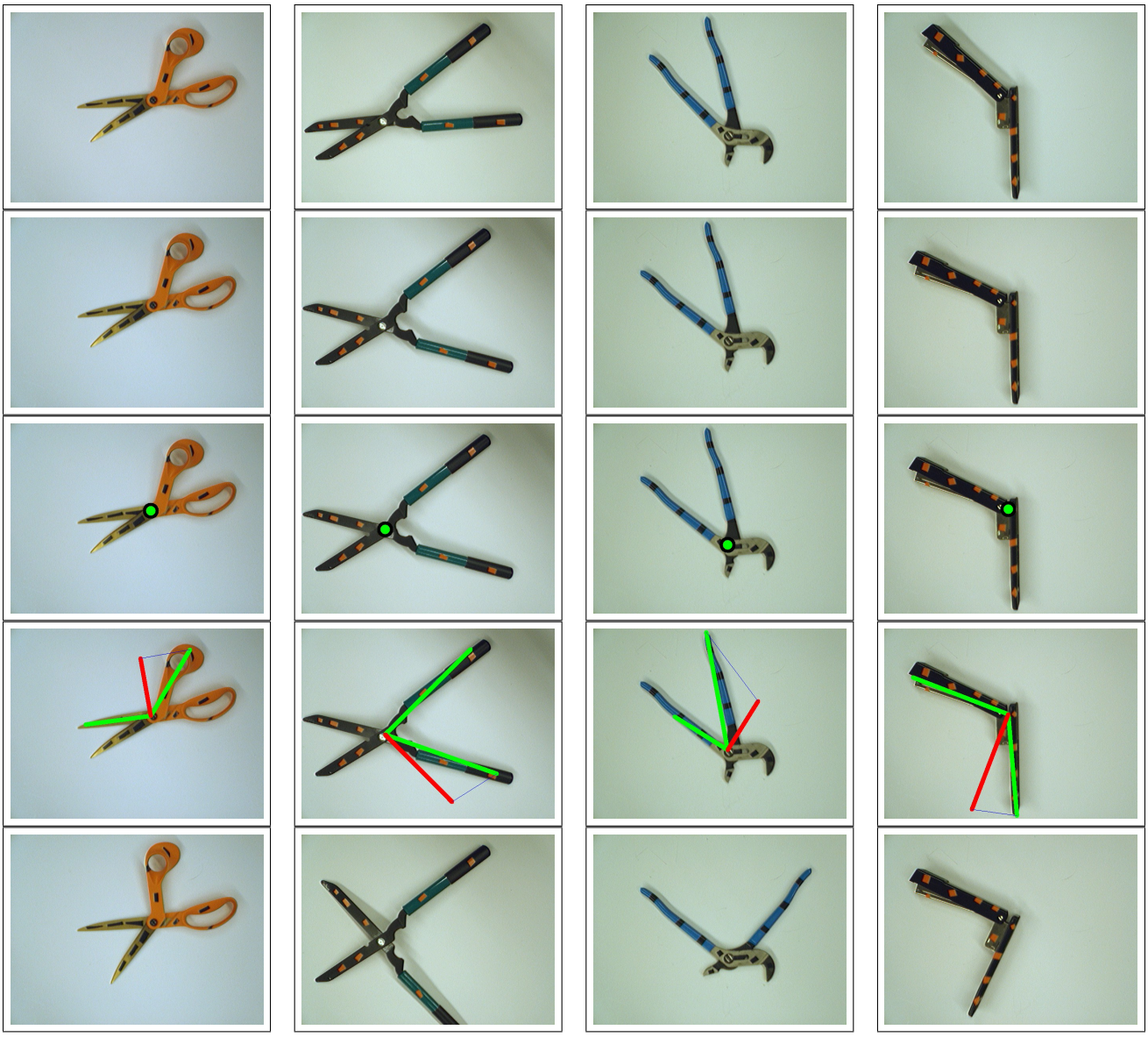

Fig. 5. Experimental results showing the use of the interactive perception framework in extracting a model of the kinematic properties of different objects. The first row of images shows the four objects (scissors, shears, plier, and stapler) in their initial pose. The second row shows the final pose of the four objects after the robot has interacted with them. The third row shows the revolute joint that was detected using the methods described in this paper; the revolute joint is marked with a green circle. The fourth row of images shows the links of the obtained kinematic model and the manipulation plan to form a right angle between the two links of the tools. Putting the two links into a 90° angle here serves as an example of tool use. The links of the tools are shown as green lines, and the orientation of one of the links to achieve the goal configuration of the tool is marked by a red line. The last row of images shows the results of executing the manipulation plan as presented in the previous row: the two links of the tools have been arranged in a 90° angle.

For example, the scissors are much smaller than the shears, have different handles and different colors. The pliers have very long handles compared to the size of their teeth. And finally, the stapler's links do not extend to both sides of the joint, unlike the other three tools. Despite these differences in appearance, all four tool belong to the family of two-link kinematic chains with a single revolute joint.

Figure 5 illustrates the experiments done using four tools: scissors, shears, plier, and a stapler. Each row represents a different phase in our algorithm. First, we see the tools in their initial pose (before the interaction begins). Next, we see the tools in their final pose (after the interaction). The third row shows the location of the axis of rotation (joint), as detected by interacting with the tools. In the fourth row, two straight lines mark the position of the links, and a third line indicates where one of the links needs to be moved to in order to create a right angle between the links. Finally, the last image shows the tools after the execution of the plan from the previous image — each tool was manipulated to form an angle of 90°.

The images in the third row of Figure 5 shows the revolute joint that was detected using the algorithm described in this paper. The axis of rotation is marked by a neon green circle. In all four cases the detection is very accurate.

The images in the fourth row of Figure 5 shows the detected links (marked by straight neon green lines). The algorithm described earlier uses the information collected in the interaction with the tool to build a kinematic model. The kinematic model was queried with the task of forming a right angle between the links of each tool. The red straight line marks the plan suggested by the model, that is how the relevant link needs to be positioned to form an angle of 90° between the two links.

Finally, the images in the last row of Figure 5 shows the four tools after executing the plan predicted by the kinematic model (depicted in the previous row). The task was to form a right angle between the links of each tool, and the results are very accurate in all cases.

The performance of our algorithm holds promise for future instances of Interactive Perception. We find that the accurate results that the robot achieved despite the simplicity of the algorithm and the the ease of implementation are very encouraging. Moreover, the implementation itself requires no parameter tuning. All experiments were done using the same executable, albeit the objects had different sizes and shapes, and the distance between the objects and the camera was changed.

VI. Conclusion

We introduced Interactive Perception as a perceptual paradigm for autonomous robots. This new framework does not distinguish between manipulation and sensing, attempting to close the gap between sensing and acting. High degree of integration between action and perception increases the robustness of the system as it allows it to handle a variety of unexpected scenarios in dynamically changing environments. Interactive Perception enables a robot to actively improve the information collected from its sensors by interacting with its environment in a directed fashion. In turn, this focused knowledge is used to improve future interaction by building better models and plans.

In this paper we have given a specific example of a complete action-perception cycle. We have shown that a robot can easily extract the kinematic properties of novel objects from a visual sensor stream if it is able to physically interact with these objects. We have further demonstrated how the extracted knowledge about the object can be used to determine appropriate use of the object. By interacting with the world, the robot extracts information about the world, which in turn enables it to interact with the world ways that facilitate the accomplishment of a specific task. Action and perception are integrated into a continuous synergetic cycle.

We hope that this work demonstrates the necessity for research in computer vision and in robotics to become more integrated. We further hope to have demonstrated that Interactive Perception is a promising perceptual paradigm for autonomous robotics.

Interactive Perception is a newly emerging field, and as such there are many possible directions for future research and extensions of the presented work. In our future work, we will remove the simplifying assumptions introduced during our experiments. This includes improving our feature tracking and contour detection algorithms by using active segmentation techniques [9], [21]. A second direction we would like to explore is the extendability of our algorithm to general kinematic chains. We thus need to extend our algorithm to also recognize other types of joints, such as prismatic joints or spherical joints. The algorithm further needs to be extended to handle objects that contain multiple separate joints. Another interesting extension to this work would be adding the capability to analyze kinematic chains that have more than one joint.

We plan to extend the framework of Interactive Perception to include more sensors (i.e. force sensors, laser scanners, multiple cameras). With the integration of multiple sensors and actuation, we anticipate that the Interactive Perception framework will enable us to address many exciting problems. One example is the task of opening door (Figure 5), which involves learning the kinematics of doors and door knobs as well as the amount of force required to twist a handle and push the door.

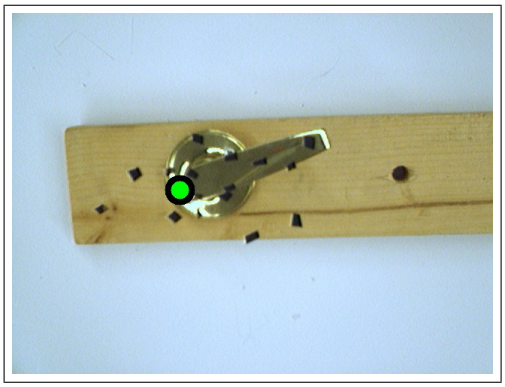

Fig. 6. The axis of rotation for a door knob, extracted using Interactive Perception.

VII. Acknowledgments

This work is supported in part by the National Science Foundation (NSF) under grants CNS-0454074, IIS-0545934, CNS-0552319 CNS-0647132, and by QNX Software Systems Ltd. in the form of a software grant. We are grateful for this support. We would also like to thank our lab manager Emily Horrell.

References

[1] Aloimonos J. and Weiss I. and Bandyopadhyay A. Active Vision. In *1st International Conference On Computer Vision*, pages 35–54, 1987.

[2] R. Bajcsy. Active perception. *IEEE Proceedings*, 76(8):996–1006, 1988.

[3] A. Blake and A. Yuille. *Active Vision*. The MIT Press, 1992.

[4] A. D. Christiansen, M. Mason, and T. Mitchell. Learning reliable manipulation strategies without initial physical models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1224–1230, 1990.

[5] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[6] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical methods. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[7] J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[8] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *Proceedings of the Robotics Science and Systems Conference*, 2006.

[9] Fedora. http://fedora.redhat.com/.

[10] P. Fitzpatrick and G. Metta. Grounding vision through experimental manipulation. *Philosophical Transactions of the Royal Society: Mathematical, Physical, and Engineering Sciences*, 361(1811):2165–2185, 2003.

[11] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.

[12] M. Hasenjäger and H. Ritter. Active learning with local models. *Neural Processing Letters*, 7(2):107–117, 1998.

[13] R. Holmberg and O. Khatib. Development and control of a holonomic mobile robot for mobile manipulation tasks. *International Journal of Robotics Research*, 19(11):1066–1074, 2000.

[14] B. K. P. Horn. *Robot Vision*. The MIT Press, 1986.

[15] S. A. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Trans. Robotics and Automation*, 12(5):651–670, October 1996.

[16] Intel. http://www.intel.com/technology/computing/opencv/.

[17] D. Katz, E. Horrell, Y. Yang, B. Burns, T. Buckley, A. Grishkan, V. Zhylkovskyy, O. Brock, and E. Learned-Miller. The UMass Mobile Manipulator UMan: An Experimental Platform for Autonomous Mobile Manipulation. In *Workshop on Manipulation in Human Environments at Robotics: Science and Systems*, 2006.

[18] J. J. Koederink and A. J. Van Doorn. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta.*, 22:773–791, Sept. 1975.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 20, pages 91–110, 2003.

[20] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision (darpa). In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, April 1981.

[21] S. Maybank. The angular velocity associated with the optical flowfield arising from motion through a rigid environment. In *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, volume 401, pages 317–326, Oct 1985.

[22] G. Metta and P. Fitzpatrick. Early integration of vision and manipulation. *Adaptive Behavior*, 11(2):109–128, 2003.

[23] A. Natsev. Multimodal Search for Effective Video Retrieval. In *International Conference on Image and Video Retrieval*, pages 525–528, 2006.

[24] QNX. http://www.qnx.com.

[25] N. Rasiwasia, N. Vasconcelos, and P. J. Moreno. Query by Semantic Example. In *International Conference on Image and Video Retrieval*, 2006.

[26] A. Stoytchev. Behavior-grounded representation of tool affordances. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3071–3076, 2005.

[27] W. T. Townsend and J. K. Salisbury. Mechanical design for whole-arm manipulation. *Robots and biological systems: towards a new bionics?*, pages 153–164, 1993.

[28] A. M. Waxman and S. Ullman. Surface structure and three-dimensional motion from image flow kinematics. *The International Journal of Robotics Research*, 4:72–94, 1985.