Robot Manipulation of Human Tools: Autonomous Detection and Control of Task Relevant Features

Charles C. Kemp
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts
cckemp@csail.mit.edu

Aaron Edsinger
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts
edsinger@csail.mit.edu

***Abstract*— The efficient acquisition and generalization of skills for manual tasks requires that a robot be able to perceive and control the important aspects of an object while ignoring irrelevant factors. For many tasks involving everyday tool-like objects, detection and control of the distal end of the object is sufficient for its use. For example, a robot could pour a substance from a bottle by controlling the position and orientation of the mouth. Likewise, the canonical tasks associated with a screwdriver, hammer, or pen rely on control of the tool's tip. In this paper, we present methods that allow a robot to autonomously detect and control the tip of a tool-like object. We also show results for modeling the appearance of this important type of task relevant feature.** [1]

I. Introduction

Robots that manipulate everyday objects in unstructured, human settings could more easily work with people and perform tasks that are important to people. We would like robots to autonomously acquire task knowledge within this context. Approaches that rely on detailed representations of specific objects and tasks are difficult to generalize to novel objects and settings. Ideally, a robot would encode task knowledge in terms of task relevant features that are important to its goal and are invariant across specific objects.

An important class of task relevant features is the tip of a tool. In this paper, we describe an approach for the autonomous detection and control of the tip of an unknown tool-like object that is rigidly grasped by a robot. For a wide variety of human tools, control of the tool's endpoint is sufficient for its use. For example, use of a screwdriver requires precise control of the position and force of the tool blade relative to a screw head but depends little on the details of the tool handle and shaft. Radwin and Haney [17] describe 19 categories of common power and hand tools. Approximately 13 of these tool types include a distal point that can be considered the primary interface between the tool and the world.

The prevalence of this type of feature may relate to the advantages it gives for perception and control. For perception, it improves visual observation of the tool's use by reducing occlusion, and it assists force sensing by constraining the interaction forces to a small region. For control, its distal location increases maneuverability by reducing the possibility of collisions. A single tip also defines the tool's interface to the world as a simple, salient region. This allows the user to perceptually attend to and control a single artifact, which reduces cognitive load. Looking beyond human tools, one can also find this structure in the hand relative to the arm, and the finger tip relative to the finger.

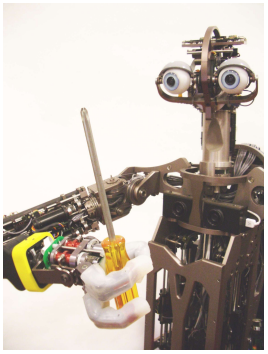

Fig. 1. Domo, the robot with which we obtained our results.

Focusing on a task relevant feature, such as the tip of a tool, is advantageous for task learning. In the case of tool use, it emphasizes control of the tool rather than control of the body. This could allow the system to generalize what it has learned across unexpected constraints such as obstacles, since it does not needlessly restrict the robot's posture. It also presents the possibility of generalizing what it has learned across manipulators. For example, a tool could be held by the hand, the foot, or the elbow and still used to achieve the same task by controlling the tip in the same way. Additionally, the function of the tip is often shared across related tools, and task knowledge could potentially be transferred between objects.

We have previously presented a method that uses the maximum point of optical flow to detect the tip of an

[1]This work was sponsored by the NASA Systems Mission Directorate, Technical Development Program under contract 012461-001.

Fig. 2. We previously demonstrated the tip detection on these tools. (hot-glue gun, screwdriver, bottle, electrical plug, paint brush, robot finger, pen, pliers, hammer, and scissors). The method performed best on the tools with sharp tips.

unmodeled tool and estimate its 3D position with respect to the robot's hand [10]. In this approach, the robot rotates the tool while using optical flow to detect the most rapidly moving image points. It then finds the 3D position of the tip with respect to its hand that best explains these noisy 2D detections. The method was shown to perform well on the wide variety of tools pictured in Figure 2. However, the detector was specialized for tools with a sharp tip, which limited the type of objects that could be used.

In this paper, we extend this work in two ways. First, we present a new multi-scale motion-based feature detector that incorporates shape information. This detector performs well on objects that do not have a sharp point, allowing us to expand our notion of the tip of an object to include such items as a bottle with a wide mouth, a cup, and a brush. The bottle and the cup are not tools in a traditional sense, yet they still have a tip or endpoint that is of primary importance during control. We show that this new feature detector outperforms our previous method on these three objects and that the estimated position and scale of the tip can be used to build a visual model. Second, we describe a method for control of the position and orientation of the tool in the image. We show results on the humanoid robot (Figure 1) described in [3], using an integrated behavior system that first performs tip detection and estimation, and then uses open-loop control to servo the tool in the image to a target location and orientation.

II. Related Work

Human use of task relevant features has been explored experimentally and theoretically. In particular, Todorov and Jordan [19] suggest that motor coordination may be optimized in terms of the task objectives rather than detailed motor trajectories. Work involving robot manipulation of task relevant features typically involves fiducial markers or simple objects. Jagersand and Nelson [7] have demonstrated that many tasks can be visually planned and executed using sparse, task relevant, fiducial markers placed on objects. Piater and Grupen [15] showed that task relevant visual features can be learned to assist with grasp preshaping. The work was conducted largely in simulation using planar objects, such as a square and triangle. Pollard and Hodgins [16] have used visual estimates of an object's center of mass and point of contact with a table as task relevant features for object tumbling. While these features allowed a robot to generalize learning across objects, the perception of these features required complex fiducial markers.

Research involving robot tool use often assumes a prior model of the tool or constructs a model using complex perceptual processing. A recent review of robot tool use finds few examples of robots using human tools [18]. NASA has explored the use of human tools with the Robonaut platform, which has used detailed tool templates to successfully guide a standard power drill to fasten a series of lugnuts [6]. Approaches that rely on the registration of detailed models are not likely to efficiently scale to the wide variety of human tools. Williamson [20] demonstrated robot tool use in rhythmic activities such as drumming, sawing, and hammering by exploiting the natural dynamics of the tool and arm. This work required careful setup and tools that were rigidly fixed to the hand.

The robot hand can be thought of as a specialized type of tool, and many researchers have created autonomous methods of visual hand detection through motion including [4] and [14]. These methods localize the hand or arm, but do not select the endpoint of the manipulator in a robust way.

In the work of Brooks [1], perception is directly coupled to action in the form of modular behaviors that eschew complex intermediate representations. Our method relates to this work in three ways. First, the robot's action is used to simplify the perceptual problem. Second, the method directly detects and controls the tip of the tool without requiring a complex representation. Third, our approach is suitable for implementation as a real-time, modular behavior.

With respect to the computer vision literature, our tip detector is a form of spatio-temporal interest point operator that gives the position and scale that are likely to correspond with the moving tool tip [13]. A similar algorithm was presented by Kemp in [9]. The multi-scale histograms generated by the detector (Figure 4) have similarities to the output from classic image processing techniques such as the distance transform, medial axis transform, and Hough transform for circles [5].

III. Review of Basic Tip Detection and Estimation

In this section we summarize the basic tool tip detection method, which we describe in detail within [10]. Our approach consists of two components. First, a tool tip detector finds candidate 2D tool tip positions within the image while the robot rotates the tool within its grasp. Second, a generative probabilistic model is used to estimate the 3D position of the tool tip within the hand's coordinate system that best accounts for these 2D detections.

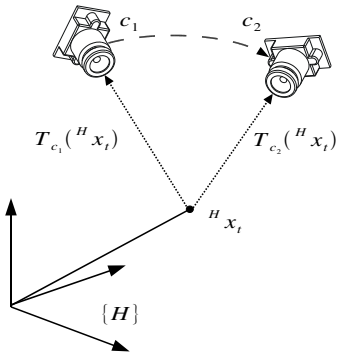

Fig. 3. The geometry of the tool tip 3D estimation problem. With respect to the hand's coordinate system, $\{H\}$, the camera moves around the hand. In an ideal situation, only two distinct 2D detections would be necessary to obtain the 3D estimate. Given two observations with kinematic configurations c_1 and c_2, the tool tip, ${}^H x_t$, appears in the image at $T_{c_1}({}^H x_t)$ and $T_{c_2}({}^H x_t)$.

A. Tip Detection

We wish to detect the 2D image position of the end point of a tool in a general way. This 2D detection can be noisy since the 3D position estimation that follows uses the kinematic model to filter out noise and combine detections from multiple 2D views of the tool.

The 2D tip detector looks for points that are moving rapidly while the hand is moving. This ignores points that are not controlled by the hand and highlights points under the hand's control that are far from the hand's center of rotation. Typically, tool tips are the most distal component of the tool relative to the hand's center of rotation, and consequently have higher velocity. The hand is also held close to the camera, so projection tends to increase the speed of the tool tip in the image relative to background motion.

In our initial work, the tool tip detector returned the location of the edge pixel with the most significant motion relative to a global motion model. In this paper, we use the same optical flow algorithm to compute the significance of an edge's motion, but perform multi-scale processing on a motion-weighted edge map to detect the tool tip.

As described in detail within [10], the optical flow computation first uses block matching to estimate the most likely motion for each edge and a 2D covariance matrix that models the matching error around this best match. Next, a global 2D affine motion model is fit to these measurements. Finally, the significance of the motion for each edge is computed as the Mahalanobis distance between the edge's measured motion model and the global motion model. This motion measurement incorporates both the magnitude of the edge's motion and the uncertainty of the measurement.

B. 3D Estimation

After acquiring the 2D tip detections in a series of images with distinct views, we use the robot's kinematic model to combine these 2D points into a single 3D estimate of the tool tip's position in the hand's coordinate system. To do this, we use the same 3D estimation technique described in [10], which we summarize here.

With respect to the hand's coordinate system, $\{H\}$, the camera moves around the hand while the hand and tool tip remain stationary. This is equivalent to a multiple view 3D estimation problem where we wish to estimate the constant 3D position of the tool tip, x_t, with respect to $\{H\}$ (For clarity we will use x_t to denote the tip position in the hand frame ${}^H x_t$). In an ideal situation, only two distinct 2D detections would be necessary to obtain the 3D estimate, as illustrated in Figure 3. However, we have several sources of error, including noise in the detection process and an imperfect kinematic model.

We estimate x_t by performing maximum likelihood estimation with respect to a generative probabilistic model. We model the conditional probability of a 2D detection at a location d_i in the image i with the following mixture of two circular Gaussians,

$$p(d_i|x_t, c_i) = (1-m)\mathcal{N}_t(T_{c_i}(x_t), \sigma_t^2 I)(d_i) + m\mathcal{N}_f(0, \sigma_f^2 I)(d_i). \quad (1)$$

$\mathcal{N}_t$ models the detection error dependent on x_t with a 2D circular Gaussian centered on the true projected location of the tool tip in the image, $T_{c_i}(x_t)$. T_{c_i} is the transformation that projects the position of the tool tip x_t onto the image plane given the configuration of the robot, c_i. T_{c_i} is defined by the robot's kinematic model and camera model. $\mathcal{N}_f$ models false detections across the image that are independent of the location of the tool tip. $\mathcal{N}_f$ is a 2D Gaussian centered on the image with mean 0 and a large variance σ_f. m is the mixing parameter.

Assuming that the detections over a series of images, i, are independent and identically distributed, and that the position of the tip, x_t, is independent of the series of configurations $c_1 \dots c_n$, the following expression gives the maximum likelihood estimate for x_t,

$$\widehat{x_t} = \text{Argmax}_{x_t} \left(\log\left(p(x_t)\right) + \sum_i \log\left(p(d_i|x_t, c_i)\right) \right) \quad (2)$$

We define the prior, $p(x_t)$, to be uniform everywhere except at positions inside the robot's body or farther than 1 meter from the center of the hand. We assign these unlikely positions approximately zero probability. We use the Nelder-Mead Simplex algorithm implemented in the open source SciPy scientific library to optimize $\widehat{x_t}$ with respect to this cost function [8].

IV. Interest Point Detection

In our original approach we modeled the tip of a tool as a single point within the image. Here we extend this approach by modeling the tip of a tool as occupying a circular area of

Fig. 4. An example of the set of 2D histograms, m_s, produced by the interest point detector when given a rectangle of edges weighted equally with unit motion. The scale, s, increases from left to right. Strong responses in the planes correspond with corners, parallel lines, and the ends of the rectangle.

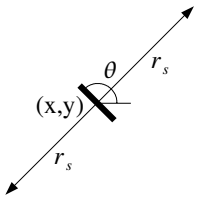

Fig. 5. This figure depicts the approximate locations in the image of the two votes at scale s cast by an edge with orientation θ and position (x, y).

some radius. In this section we describe this extension, which has better performance on tools with tips that do not come to a sharp point. Since this new estimate includes the spatial extent of the tip, it also facilitates the use of visual features that describe the appearance of the tip over this spatial extent. For example, given the position and radius we can collect appropriately scaled image patches, see Figure 7.

With respect to our goal of detecting the tip of a tool, this detector implicitly assumes that the end of an object will consist of many strongly moving edges that are approximately tangent to a circle at some scale. Consequently, the detector will respond strongly to parts of the object that are far from the hand's center of rotation and have approximately convex projections onto the image. As our results show, the detections correspond well with human-labeled tips.

The input to the interest point detector consists of a set of weighted edges, e_i, where each edge i consists of a weight, w_i, an image location, x_i, and an angle, θ_i. We use a Canny edge detector to produce edge locations and orientations, to which we assign weights that are equal to the estimated motion. Each edge votes on locations in a scale-space that correspond with the centers of the coarse circular regions the edge borders. For each edge, we add two weighted votes to the appropriate bin locations at each integer scale s.

As depicted in Figure 5, within the original image coordinates the two votes are approximately at a distance r_s from the edge's location and are located in positions orthogonal to the edge's length. We assume that the angle θ_i denotes the direction of the edge's length and is in the range $[-\frac{\pi}{2}, \frac{\pi}{2})$, so that no distinction is made between the two sides of the edge.

For each scale s there is a 2D histogram that accumulates votes for interest points. The planar discretization of these histograms is determined by the integer bin length, l_s, which is set with respect to the discretization of the scale-space over scale, $l_s = \lceil \beta(r_{s+0.5} - r_{s-0.5}) \rceil$, where β is a scalar constant that is typically close to 1.

We define r_s such that r_{s+1} is a constant multiple of r_s, where s ranges from 1 to c inclusive. We also define r_s to be between r_{max} and r_{min} inclusive, so that

$$r_s = \exp\left(\frac{\log(r_{max}) - \log(r_{min})}{c-1}(s-1) + \log(r_{min})\right) \quad (3)$$

Setting r_{min} and r_{max} determines the volume of the scale-space that will be analyzed, while c determines the resolution at which the scale-space will be sampled.

We compute the bin indices, (b_x, b_y), for the 2D histogram at scale s with

$$b_s(x, \theta) = \text{round}\left(\frac{1}{l_s}\left(x + r_s \begin{bmatrix} \cos(\theta + \frac{\pi}{2}) \\ \sin(\theta + \frac{\pi}{2}) \end{bmatrix}\right)\right), \quad (4)$$

which adds a vector of length r_s to the edge position x and then scales and quantizes the result to find the appropriate bin in the histogram.

We now iterate through the edges adding their weighted contributions to the appropriate bins. We can write the equation for the resulting interest point detection maps, m_s, using delta functions, δ, so that

$$m_s(u) = \sum_i w_i(\delta(u - b_s(x_i, \theta_i)) + \delta(u - b_s(x_i, \theta_i + \pi))), \quad (5)$$

where $\delta(x) = \begin{cases} 1 \text{ if } (x_x = 0) \wedge (x_y = 0) \\ 0 \text{ otherwise} \end{cases}$.

In order to soften the effects of our block discretization, we low-pass filter each 2D histogram, m_s, with a separable, truncated, FIR Gaussian, which is approximately equal to giving each edge a Gaussian vote distribution, since

$$G \star m_s = \sum_i w_i(G(u - b_s(x_i, \theta_i)) + G(u - b_s(x_i, \theta_i + \pi))), \quad (6)$$

where G is an ideal Gaussian. This is also approximately equal to blurring the weighted edge map by scale varying Gaussians, or blurring the scale-space volume across scale.

Ideally, the values of corresponding interest points resulting from a shape would be invariant to translation, scaling, and rotation of the shape. We introduce two scalar functions n_s and n_θ to reduce scale dependent variations and angle dependent variations respectively, so that

$$m_s(u) = n_s \sum_i n_{\theta_i} w_i (G(u - b_s(x_i, \theta_i)) + G(u - b_s(x_i, \theta_i + \pi))). \quad (7)$$

We determine the values for these two functions empirically using a calibration pattern.

Finally, we find the point within the scale-space with the highest response and use its corresponding position and scale within the image as the 2D tip detection.

V. Control of the Tool in the Image

As described in Section III, we use the 2D tip estimates to produce a 3D estimate of the tip's location within the hand's coordinate frame. This effectively extends our kinematic model and provides many options for controlling the tip. In this section we consider the control of the position and orientation of this task-relevant feature within the visual image.

As a step toward visual servo control of the tip, we tested a feedforward approach for control of the tool. The approach is a variant of the well studied area of resolved-rate motion control [12] and operational-space control [11]. The robot used in this paper, seen in Figure 1, has 4 DOF in the arm, 2 DOF in the wrist, and 7 DOF in the head.

A kinematic model of the head and arm is known. We also know the camera's intrinsic parameters and remove radial distortion from the image. The head remains fixed and therefore ${}^{W}_{I}T$, the transform between world coordinates and image coordinates, is constant.

A Jacobian transpose approach allows us to minimize the error between the desired tool pose and the estimated pose, if the joint angles start close to their final state [2]. The Jacobian, ${}^{W}J^{T}$, is known from the kinematic model and relates hand forces to joint torques as $\tau = {}^{W}J^{T}\ {}^{W}f$. Instead of controlling the arm's joint torque directly, we control the joint angle, and our controller takes the form of $\Delta\theta = \sigma\ {}^{W}J^{T}\ {}^{W}f$ for controller gains σ.

The position and orientation of the tip is controlled through simulated forces, ${}^{W}f$, created by virtual springs in the hand's coordinate frame $\{H\}$. One virtual spring controls the position of the tip by connecting the estimated position of the tip, ${}^{H}x_t$, with the target location, ${}^{H}x_d$. The other virtual spring controls the orientation of the tip by connecting the estimated position of the robot's hand, ${}^{H}x_p$, with a target location ${}^{H}x_o$. The target locations for the tip and the hand are constrained to lie at a fixed depth along the camera's optical axis. The virtual forces acting at the hand are then:

$${}^{H}f_t = {}^{H}J^{T} \begin{bmatrix} ({}^{H}x_d - {}^{H}x_t) & 0 & 0 & 0 \end{bmatrix}^{T} \quad (8)$$

$${}^{H}f_p = {}^{H}J^{T} \begin{bmatrix} ({}^{H}x_o - {}^{H}x_p) & 0 & 0 & 0 \end{bmatrix}^{T}. \quad (9)$$

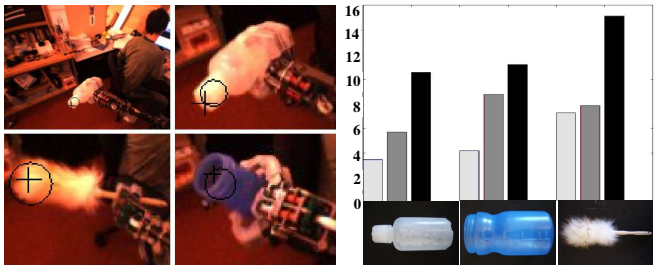

Fig. 6. Left: The upper left image gives an example of the images used during estimation. The movement of the person in the background serves as a source of noise. In the other three images the black cross marks the hand annotated tip location and has a width equivalent to twice the mean pixel error for prediction over the test set. The black circle is at the tip prediction with a size equal to the average feature scale. Right: The mean prediction error, in pixels, for each tool. The 3D tool pose is estimated in three ways: the hand labelled tool tips [left bar], feature-based interest points [middle bar], and the edge pixel with the maximum motion [right bar].

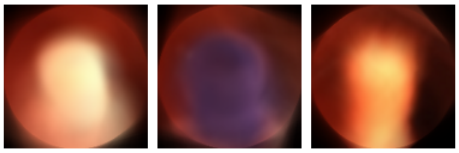

Fig. 7. These average tip images give an example of acquiring a model of the tip's appearance. Square image patches of the tips were collected using the tip detector, tip predictor, and smoothing of the estimated state. They were then normalized in scale and orientation and averaged together.

where ${}^{H}J^{T}$ relates forces in $\{H\}$ to a wrench at the hand. We can transform forces from frame $\{H\}$ to $\{W\}$ through:

$${}^{W}_{H}J^{T} = \begin{bmatrix} {}^{W}_{H}R & 0 \\ 0 & {}^{W}_{H}R \end{bmatrix}. \quad (10)$$

giving ${}^{W}f_t = {}^{W}_{H}J^{T}\ {}^{H}f_t$ and ${}^{W}f_p = {}^{W}_{H}J^{T}\ {}^{H}f_p$, where ${}^{W}_{H}R$ is the rotational component of ${}^{W}_{H}T$. A spherical 3 DOF wrist would allow decoupling of the control problem into position control by the arm and orientation control by the wrist, giving the controllers:

$$\Delta\theta_{wrist} = {}^{W}J^{T} \left(\sigma_{twrist}\ {}^{W}f_t + \sigma_{pwrist}\ {}^{W}f_p\right) \quad (11)$$

$$\Delta\theta_{arm} = {}^{W}J^{T} \left(\sigma_{tarm}\ {}^{W}f_t + \sigma_{parm}\ {}^{W}f_p\right) \quad (12)$$

for controller gains σ. The wrist used in our experiments has only 2 DOF and consequently we must ignore the third joint and assume that the correct orientation is locally achievable with the restricted kinematics. These decoupled controllers will bring the estimated tool pose into alignment with a desired pose if the controller is initialized at a joint pose near the final solution.

VI. Results

We validated our method on a bottle, a cup, and a brush, as pictured in Figure 6. The items were chosen for their varying

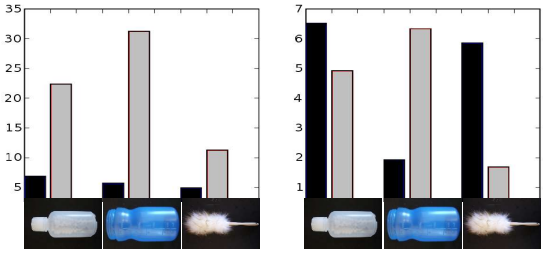

Fig. 8. The controller error for a single trial for the tip position [pixels] and orientation [degrees] using the multi-scale detector. The controller error [black] is measured by the projection of the predicted tip position and orientation into the image. The hand measured [grey] position and orientation error are also measured. The relative errors for the orientation results, $(7degrees)$, were too small to be measured with precision.

tip size and length. The feature detector, estimator, and controller were integrated into a real-time behavior module for the robot. The detection algorithm runs at $15Hz$ on a $3GHz$ Pentium computer. It is run in parallel for the robot's two cameras. When the tool is placed in the robot's hand, it automatically generates a short sequence of tool motion of about 200 samples over 5 seconds. Each detection and kinematic configuration is logged and then batch processed by the estimator. The estimated tip location, ${}^{H}x_t$, is passed to the tool pose controller and it servos the tool to a potentially time-varying location and orientation in the image.

For each tool we compare the multi-scale detector of this paper to the original edge-motion detector. Figure 6 shows the mean prediction error, as measured by the tool tip projection into the image, for the two detectors. The multi-scale detector significantly improves the predicted location for these three objects that have large, broad tips. The detector also enables online modeling of the tip. Figure 6 shows the average estimated tip scale for each tool, which demonstrates the ability of the detector to appropriately extract the size of the tool tip. Figure 7 illustrates the construction of a pose normalized visual model of the tip.

We tested the tool tip controller by servoing the tip of each tool to the center of the image with a horizontal orientation. Figure 8 shows the typical errors, relative to the predicted tip location in the image, and relative to the actual hand labelled tip location. The controller is able to successfully bring the estimated (though not necessarily the actual) tip location to the desired pose. In the future we plan to incorporate visual feedback based on the tip's visual model to reduce errors accumulated from kinematic inaccuracies.

Our work affords many avenues for further exploration. The reliable prediction of the tool tip in the visual scene allows us to model the tool's visual features, which should enable us to visually track the tip and actively test and observe the endpoint during task execution. Additionally, the approach should be applicable to skill transfer from a human to a robot based on observation of the tool tip rather than the kinematic details of the task.

We have described a general method for visual manipulation of human tools rigidly held by a robot. It is a step towards robots that autonomously learn to perform manipulation tasks with novel, unmodeled objects in human-centric environments.

REFERENCES

[1] Rodney A. Brooks. *Cambrian Intelligence*. MIT Press, Cambridge, MA, 1999.

[2] J. Craig. *Introduction to Robotics*. Addison Wesley, 2 edition, 1989.

[3] Aaron Edsinger-Gonzales and Jeff Weber. Domo: A Force Sensing Humanoid Robot for Manipulation Research. In *Proceedings of the 2004 IEEE International Conference on Humanoid Robots*, Santa Monica, Los Angeles, CA, USA., 2004. IEEE Press.

[4] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini. Learning About Objects Through Action: Initial Steps Towards Artificial Cognition. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, May 2003.

[5] D. A. Forsyth and Jean Ponce. *Computer Vision: a modern approach*. Prentice Hall, 2002.

[6] E. Huber and K. Baker. Using a hybrid of silhouette and range templates for real-time pose estimation. In *Proceedings of ICRA 2004 IEEE International Conference on Robotics and Automation*, volume 2, pages 1652–1657, 2004.

[7] M. Jagersand and R. Nelson. Visual Space Task Specification, Planning and Control. In *Proceedings of the IEEE International Symposium on Computer Vision*, pages 521–526, 1995.

[8] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001.

[9] Charles C. Kemp. *A Wearable System that Learns a Kinematic Model and Finds Structure in Everyday Manipulation by using Absolute Orientation Sensors and a Camera*. PhD thesis, Massachusetts Institute of Technology, May 2005.

[10] Charles C. Kemp and Aaron Edsinger. Visual Tool Tip Detection and Position Estimation for Robotic Manipulation of Unknown Human Tools. Technical Report AIM-2005-037, MIT Computer Science and Artificial Intelligence Laboratory, 2005.

[11] O. Khatib. A unified approach to motion and force control of robot manipulators: The operational space formulation. *International Journal of Robotics and Automation*, 3(1):43–53, 1987.

[12] D. Kragic and H. I. Chrisensen. Survey on visual servoing for manipulation. Technical report, Computational Vision and Active Perception Laboratory, 2002.

[13] I. Laptev. On space-time interest points. *Int. J. Computer Vision*, 64(2):107–123, 2005.

[14] Michel, Gold, and Scassellati. Motion-based robotic self-recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[15] Justus H. Piater and Roderic A. Grupen. Learning appearance features to support robotic manipulation. *Cognitive Vision Workshop*, 2002.

[16] N. Pollard and J.K. Hodgins. Generalizing Demonstrated Manipulation Tasks. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics (WAFR '02)*, December 2002.

[17] R.G. Radwin and J.T. Haney. An ergonomics guide to hand tools. Technical report, American Institutional Hygiene Association, 1996. http://ergo.engr.wisc.edu/pubs.htm.

[18] R St. Amant and A.b Wood. Tool use for autonomous agents. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 184–189, 2005.

[19] Emanuel Todorov and Michael Jordan. Optimal feedback control as as theory of motor coordination. *Nature Neuroscience*, 5(11):1226–1235, 2002.

[20] M. Williamson. *Robot Arm Control Exploiting Natural Dynamics*. PhD thesis, Massachusetts Institute of Technology, 1999.