Imirok: Real-Time Imitative Robotic Arm Control for Home Robot Applications

Heng-Tze Cheng, Zheng Sun, Pei Zhang
Department of Electrical and Computer Engineering, Carnegie Mellon University
{hengtze, zhengs, peizhang}@cmu.edu

***Abstract*—Training home robots to behave like human can help people with their daily chores and repetitive tasks. In this paper, we present Imirok, a system to remotely control robotic arms by user motion using low-cost, off-the-shelf mobile devices and webcam. The motion tracking algorithm detects user motion in real-time, without classifier training or pre-defined action set. Experimental results show that the system achieves 90% precision and recall rate on motion detection with blank background, and is robust under the change of cluttered background and user-to-camera distance.**

Keywords—Robotic arm, gesture-based human robot interaction.

I. INTRODUCTION

The interaction between human and sensor-enabled home robots in a smart home environment is important in the field of pervasive computing. While home robots have become available [1], most robots are custom-made for specific operations without the capability to meet the needs of different users. As many everyday tasks (such as cleaning, cooking, moving objects) require specific gestures and movements that vary with users and usage scenarios, it is important to design a system that allows users to train and control the robot the way they want. In this work, we focus on the research problem of how to track a user's motion and use it for controlling a robotic arm.

Recognizing user motion or gesture from low-level image signals and replay the motion on robots is technically challenging. Although extensive work has been done in gesture-based robot control [4]−[6], voice-based human robot interaction [7,8], and wearable gestural interface [9], most systems still require time-consuming training process for learning models, custom made devices or wearable equipments for controlling, and can only understand a small set of pre-defined command-action pairs. These features are undesirable for users who want to control or train their robot for custom tasks, without wearing equipments or purchasing additional devices.

In this paper, we present *Imirok*, a real-time imitative robotic arm control system shown in Figure 1, with the following contributions. First, we present the design, implementation, and evaluation of the Imirok system. Imirok provide an interface for general users to control home robots using the webcam they already have, without the need of purchasing specific hardware or high-cost instruments. Second, using an optical-flow-based motion tracking algorithm, our system can track user motion in real-time without the need of a model training process in advance. Third, rather than classifying user's motion into a pre-defined gesture set, a user can perform arbitrary motions with one's arm for the robot to imitate, which enables training robotic arm for custom tasks.

The paper is organized as follows. In Section 2, we describe the proposed motion tracking algorithm. Our approach and implementation on robotic arm control is elaborated in Section 3. The evaluation results are described in Section 4. We discuss related work in Section 5 and our conclusions in Section 6.

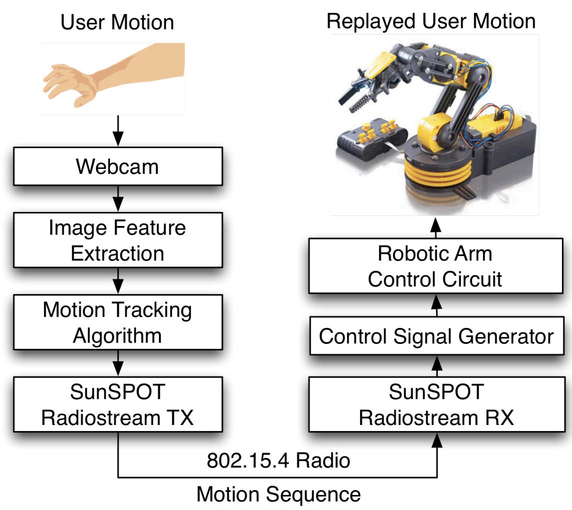

Figure 1. The system architecture of Imirok.

II. MOTION TRACKING

A. *Design Considerations*

Our goal is to infer user motions from a video stream captured by a webcam facing the user. To track the motion of the user's hand, we need to decide what kind of image features that is both robust and ideal for real-time tracking of moving points across consecutive image frames. The intuition is that if we randomly pick a point on a uniform surface to track, then we are not likely to find the same point in the next frame. Conversely, if we choose a unique point that is more invariant under motion or luminance changes, we have more chance of finding that point again and the tracking will be more robust. Also, to achieve real-time motion tracking, the feature extraction and the tracking algorithm need to be computationally efficient. These considerations motivate our approach as follows.

B. *Image Feature Extraction*

First, each input image from the video sequence is converted to grayscale image, since the color information is not required. Then, we extract the corner feature [2] since it is shown to be useful for object tracking in video. The corner feature extraction algorithm finds a list of points that are relatively useful for tracking.

C. *Optical-Flow-Based Motion Tracking*

Since we need real-time gesture recognition rather than an offline video motion analysis, we adopt Lucas-Kanade Algorithm [3], a sparse tracking method that significantly reduces computational cost compared with dense tracking.

Our approach works as follows. First, given an input frame, we extract at most N_{max} salient corner points that are suitable for tracking. Then, by comparing the current and the previous frame

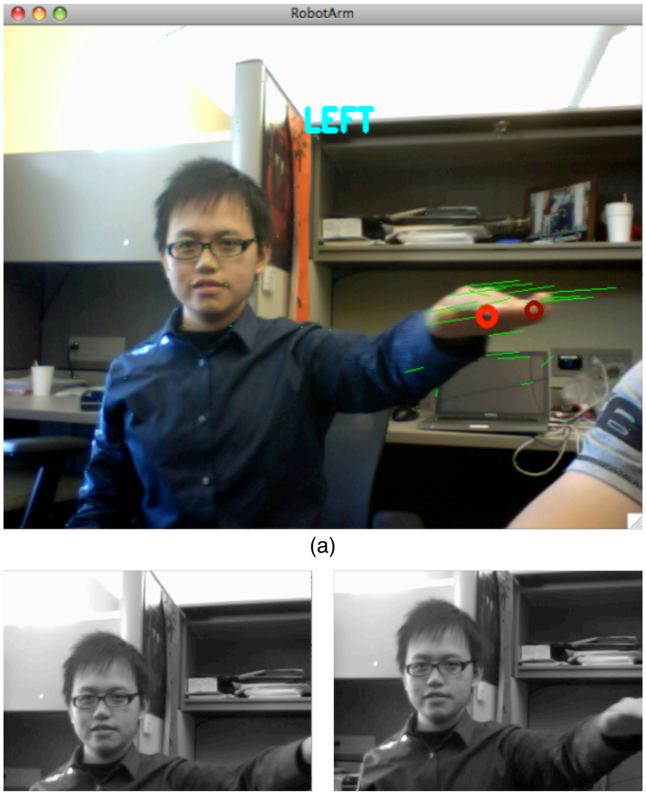

Figure 2. The screenshot (a) of the motion tracking system. The green lines connect the previous and current position of each moving points. The dark and bright red circle is the centroid of the filtered moving corners in the previous (b) and current frame (c), respectively. Even with a cluttered background, the noisy movements are filtered and the estimated hand moving direction is shown on the screen in real-time.

using Lucas-Kanade Algorithm, we obtain a set of point pairs that describes how each point moves from one frame to the next frame. In the experiments, $N_{\max}$ is empirically set to 500.

After the set of point pairs are extracted, we compute the centroid of points in the current and the previous frame, which is shown in Figure 2 with red circles. Even if the background is in a clutter of objects with various colors and shapes, as long as the user's hand is the major moving object in the scene, the centroid roughly corresponds to the hand's position.

We then compute the distance of the centroid shift. If the centroid shift is larger than a pre-determined threshold d_t in a certain direction, an estimate of moving direction is determined. Empirically, d_t is set to a distance of 10-pixel length.

III. ROBOTIC ARM CONTROL

In this paper, we present an intuitive way of controlling a robotic arm, to follow motions made by a human operator. One challenge

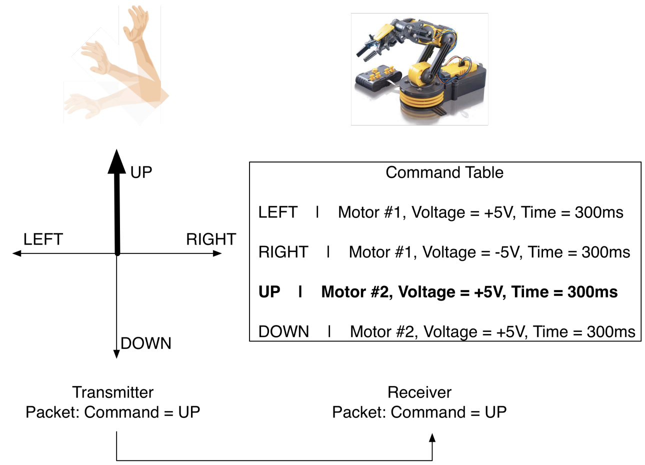

Figure 3. Motion mapping from user arm to robot arm.

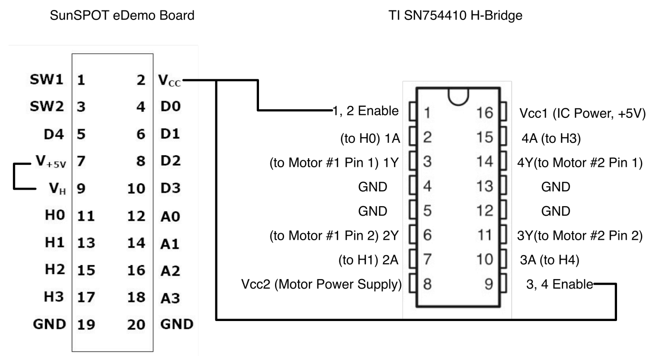

Figure 4. Motor control circuit.

is to establish a one-to-one mapping from joints of human body to sections on the robot arm. In this section, we present motion-mapping techniques in Imirok and the design of motor control circuits in the current implementation.

A. *Motion Mapping*

The object moving in front of the camera, and hence being tracked by the optical flow-based motion tracking algorithm, need not to be fixed. Any movements, which can be interpreted into the pre-defined movement table, are acceptable. For example, for issuing an "up" command, in front of the camera, an operator can raise his/her arm, head, or even shoulder. However, the motions made by the robotic arm should be fixed, so that every movement by human operators for issuing the same command should result in the same robotic arm motion.

In achieving this goal, the current Imirok adopts a table-based command lookup process. Firstly, possible movements are discretized into finite number of moving directions. These moving directions are mapped onto different sections of the robotic arm. Then, by looking up the moving directions in a predefined command table, motor control contexts, such as motor ID, voltage, and time, are provided. These contexts are then directly used in the control of the corresponding motors so that the robotic arm will make expected movements.

In Figure 3, a lookup table that illustrates a four-direction movement discretization is shown.

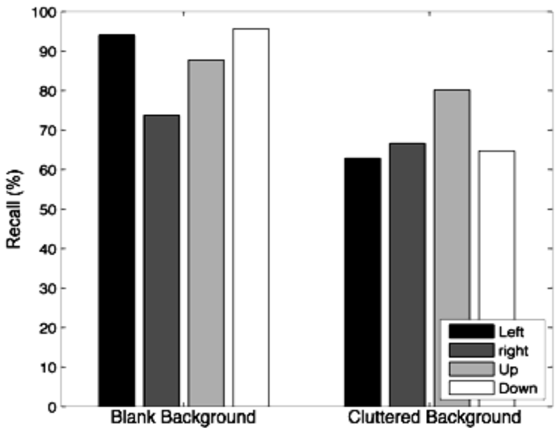

Figure 5. The effect of blank/cluttered background on the recall rate of motion tracking.

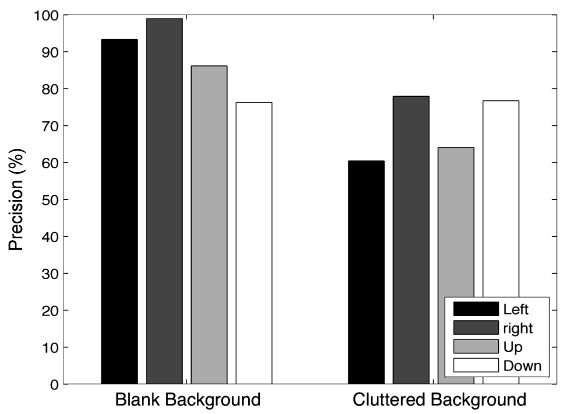

Figure 6. The effect of blank/cluttered background on the precision of motion tracking.

B. DC Motor Control

In the control of motors, H-bridges are connected to motors on the robotic arm so as to power the motors and generate different motor directions. H-bridge is an electronic device that allows DC motors to run forwards and backwards according to logical signal combinations provided [10]. Figure 4 shows the circuit of using a SunSPOT eDemo board to power and send logical signals to a H-bridge IC for controlling two motors on the robotic arm.

IV. IMPLEMENTATION

For prototype implementation, we use SunSPOT nodes for 802.15.4 radio communication between the vision-based motion tracking device (a laptop with a webcam) and the OWI-535 robotic arm [11]. TI SN754410 H-bridge IC is used for motor control. The image feature extraction and motion tracking algorithms are implemented using OpenCV library in C++.

V. EVALUATION

In this section, we present the evaluation results of motion tracking performance of Imirok, and discuss possible factors that influence the performance.

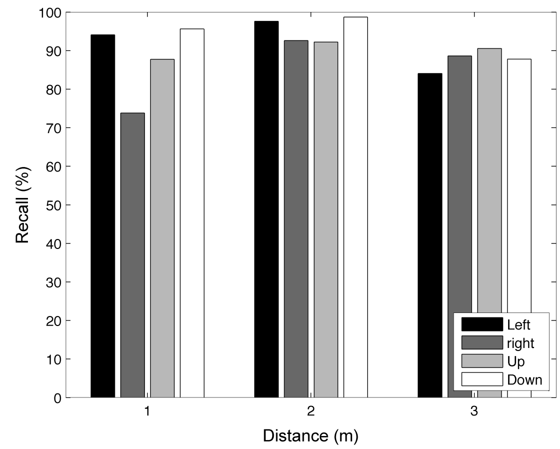

Figure 7. The effect of user-to-camera distance on the recall rate of motion tracking.

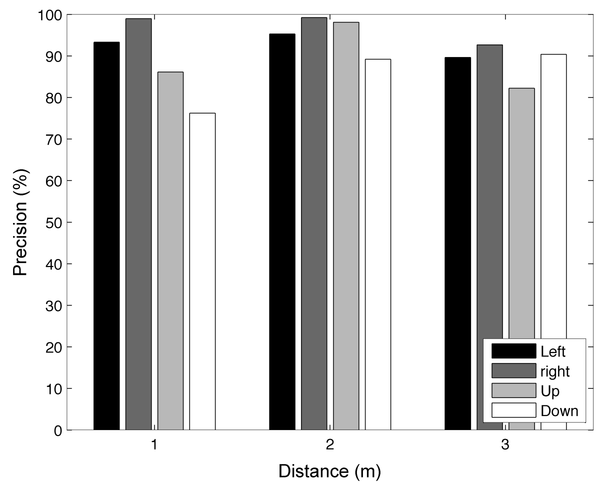

Figure 8. The effect of user-to-camera distance on the precision of motion tracking.

A. Background Complexity

Intuitively, simple backgrounds behind the human operator will generate better tracking performance than complex backgrounds having multiple objects with different colors. We evaluate the performance of Imirok under both blank and cluttered backgrounds. For the blank background, a human operator stands in front of a mono-color wall, issuing left, right, up, and down commands by moving his upper arm. For the cluttered background, the operator stands in the cubicle area in our campus. Results in Figure 5 and 6 show that blank background averagely generates 5% ~ 15% better performance of both precision and recall rates for all four types of motions.

B. Distance to Camera

The human operator stands in front of a mono-color wall with different distance to the camera. From Figure 7 and 8, it seems that the precision and recall rates of Imirok are pretty constant against distance changes. For all the cases ranging from 1m to 3m, the average precision and recall rates are over 85%.

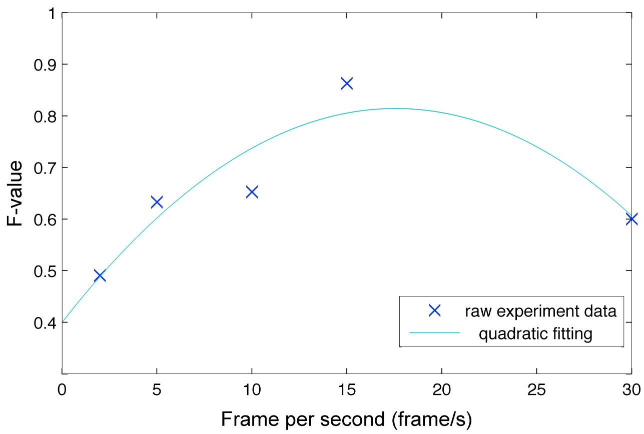

Figure 10. The effect of frame rate on the f-value of motion tracking.

C. *Frame Rate*

A promising and interesting research direction is to implement the whole Imirok system in mobile devices. A key challenge of doing so is to reduce the computation complexity. In this subsection, we compare the performance of motion tracking by changing the frame rate of the optical-flow algorithm. As shown in Figure 9, as the frame rate increases, the F-value, which is a commonly used recognition performance metric that combines precision and recall rates, increases as well. To be specific, in the experiment, when the frame rate increases from 2 to 15 frame/s, the F-value increases from 0.5 to 0.9. This is because, when the frame rate is low, Imirok tends to miss a number of motions made by the operator; as the frame rate increases, the missed motions decreases. As the frame rate keeps increasing, however, the F-value starts to drop, as observed at 30 frame/s in Figure 9. This is because, if the frame rate is too high, a movement of the human operator between adjacent images is smaller, which may be lower than the predefined threshold, so that a movement is missed. Generally speaking, a high recognition performance of Imirok requires a good selection of frame rates, threshold, and movement speeds of the human operator.

VI. RELATED WORK

There has been related work investigating gesture-based human-robot interaction. In [4,5,6], computer vision techniques, such as face detection or gesture recognition, are used to provide intuitive vision-based control interface for users. However, since most of the work relies on training specific gesture models, the robot can only understand pre-defined command-action pair. Although promising accuracy was reported on the finite command sets, a user cannot train the robot to perform customized actions for individual needs. Besides human-robot interaction, SixthSense [9] proposed a wearable gesture-based interface for mobile applications such as map navigation or photo browsing. However, a user needs to wear four color markers on the fingers for gesture recognition. In comparison, our system directly tracks a user's motion without the need of additional accessories or equipment worn on one's body.

In addition to gesture-based systems, there has also been research exploring the idea of voice control. In [7], spoken dialog is used as input commands (e.g. "go over there"). However, they focused on the simple movement of a wheeled robot similar to Roomba [1], rather than controlling robotic arms with multiple joints. In VoiceBot [8], non-verbal voice commands (e.g. "ah", "ooh", "aw") are mapped to different orientations for robotic arm control. While the work focused on the user study of a novel user interface, a detailed accuracy of control was not reported, and there is no intuitive way that a user can give an example of motion he/she want to perform.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented the design, implementation, and evaluation of Imirok, an imitative robotic arm control system. The system detect user motion in real-time, without the need of model training before using the system. The motion tracking approach achieved 90% precision and recall rate with blank background, and show acceptable robustness under the change of cluttered background and user-to-camera distance. The real-timeness, the robust performance, and the intuitive imitative human-robot interaction make the system particularly desirable for controlling home robots in a smart home environment.

There are plenty of research opportunities for future work. Our approach can be extended to increase the degree of freedom in controlling robotic arm. In addition, controlling two arms simultaneously can be enabled by adding a motion orientation clustering algorithm after motion tracking. To further enable robotic control using mobile phone cameras, it is also important to balance the tradeoff between computational cost, power consumption, transmission bandwidth, and the performance of vision-based motion tracking.

REFERENCES

[1] iRobot Corporation. [http://www.irobot.com/]

[2] J. Shi, C. Tomasi. Good Features to Track. *IEEE Conf. Computer Vision and Pattern Recognition*, 1994.

[3] J.-Y. Bouguet. Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm. *Intel Corp. Microprocessor Research Labs*, 2000.

[4] S. Waldherr, R. Romero, S. Thrun. A gesture based interface for human-robot interaction. In *Autonomous Robots*, 2000.

[5] O. Rogalla, M. Ehrenmann, R. Zollner, R. Becher and R. Dillmann. Using Gesture and Speech Control for Commanding a Robot Assistant. In *Proc. IEEE Int'l Workshop on Robot and Human Interactive Communication*, 2002.

[6] Md. Hasanuzzaman, T. Zhang,V. Ampornaramveth, H. Gotoda, Y. Shirai, H. Ueno. Adaptive visual gesture recognition for human–robot interaction using a knowledge-based software platform. In *Robotics and Autonomous Systems*, Vol. 55, 643–657, 2007.

[7] V. Kulyukin. Human-Robot Interaction Through Gesture-Free Spoken Dialogue. In *Autonomous Robots*, 2004.

[8] B. House, J. Malkin, J. Bilmes. The VoiceBot: A Voice Controlled Robot Arm. In *Proc. ACM Conf. Human Factors in Computing Systems*, 2009.

[9] P. Mistry, P. Maes. SixthSense: a wearable gestural interface. In *Proc. SIGGRAPH ASIA*, 2009.

[10] DC Motor Control Using an H-Bridge, [http://itp.nyu.edu/physcomp/Labs/DCMotorControl].

[11] OWI-535 Robotic Arm Edge: Product Information [http://www.imagesco.com/robotics/owi-535.html].

[12] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.