AN EFFICIENT METHOD FOR HUMAN POINTING ESTIMATION FOR ROBOT INTERACTION

Satoshi Ueno†, Sei Naito†, and Tsuhan Chen‡

†KDDI R&D Laboratories, Inc. ‡Cornell University

ABSTRACT

In this paper, we propose an efficient calibration method to estimate the pointing direction via a human pointing gesture to facilitate robot interaction. The ways in which pointing gestures are used by humans to indicate an object are individually diverse. In addition, people do not always point at the object carefully, which means there is a divergence between the line from the eye to the tip of the index finger and the line of sight. Hence, we focus on adapting to these individual ways of pointing to improve the accuracy of target object identification by means of an effective calibration process. We model these individual ways as two offsets, the horizontal offset and the vertical offset. After locating the head and fingertip positions, we learn these offsets for each individual through a training process with the person pointing at the camera. Experimental results show that our proposed method outperforms other conventional head-hand, head-fingertip, and eye-fingertip-based pointing recognition methods.

Index Terms— Pointing Gesture, Object Identification, Calibration, Robot Interaction

1. INTRODUCTION

Human pointing is an intuitive gesture used to indicate direction. When a user points at something, if a robot knows the plane on which the object exists (ex. floor, table, or wall), the robot can locate the object through the intersection between the pointing direction and the plane. After doing this, the robot can perform such tasks as fetching the object and discarding it. However, the identification of target objects from the user's pointing gesture is still a challenging problem for robots, sometimes even for humans. One of the intrinsic problems is how to adapt to individual ways of pointing when trying to recognize the pointing direction. For example, not all users indicate the position of the object with a careful pointing gesture that coincides with the direction of the gaze. In this case, the target object is not on the line extending forward from the dominant eye to the tip of index finger. In other words, sometimes the fingertip is not on the line from the dominant eye to the object: the line of sight. Figure 1 shows this situation. We define the offset between the target object position and the eye-fingertip line as the individual way, and it is split into two offsets, horizontal offset o_h and vertical offset o_v. If these offsets become too great to ignore, the robot will fail to identify the position of the target object even if the robot is able to extract the eye and the fingertip positions precisely. For example, when the user points to the object with a small pointing gesture [9], these offsets tend to be greater. To overcome this problem, one solution is to adapt to the pointing gesture ways by using regression methods [3, 5]. After the user points to several objects whose positions are given, the robot can estimate the pointing direction using these regression models. However, in this case, another problem arises. The problem is how

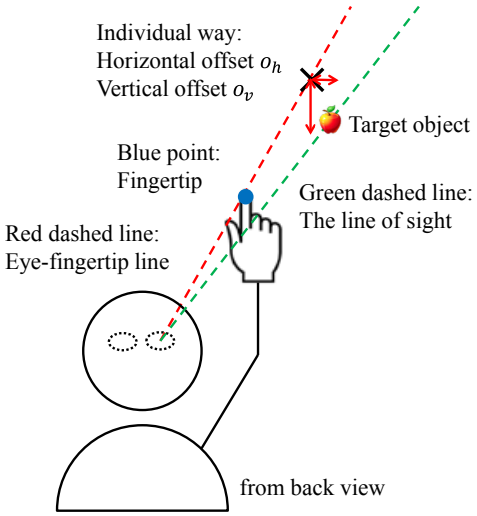


Fig. 1. Definition of offsets referred to in this paper

the robot indicates the position of the object to the user during the learning step. In the one study [5], they used screens to show the target object positions to the users. In another study [3], the users knew the positions of the objects and the users were informed of what the target objects were. In real situations, it is not an easy task for users to set up the circumstance for obtaining the learned data with some given objects.

In this paper, we propose an efficient calibration method to estimate the human pointing direction to facilitate robot interaction. Our main purpose is to adapt to individual pointing ways so that the horizontal and vertical offsets can be easily compensated for. In our preliminary experiments, we assume that the horizontal offset o_h is relatively shorter than the vertical offset o_v. Hence, we focus on how to calibrate the vertical offset for individual ways. The proposed method requires only a pointing gesture to the center of an RGB-D camera.

2. RELATED WORK

There have been many studies in the area of pointing direction recognition for robot interaction. In the last decade, some studies focused on how to detect human bodies and how to recognize pointing gestures [7, 10, 12]. However, in an indoor situation, The Microsoft Kinect and its SDK [14] have sufficient accuracy for human detection and pointing gesture detection. Thus, we can use the Kinect to detect human bodies and to recognize pointing gestures in order to focus on estimating the pointing direction.

Regarding the estimation of pointing direction, roughly speaking, the direction is approximately along the line from the user's dominant eye position to the fingertip position. However, because

it is still a difficult problem to extract these two small positions, there are many alternative methods for estimating the pointing direction; head-hand line [2, 6, 8], head-finger line [4, 6, 13], forearm direction [1, 6, 8], and head orientation [8, 11] methods. Regarding the head orientation approach, this method does not use the hand position so we need to obtain more information to identify the target object, for example, speech recognition to obtain the object features [11]. Regarding the forearm direction, it is useful to indicate the position to which the robot should go [1]. However, in studies [6, 8], it was concluded that the head-hand or head-finger methods should be used to identify the object rather than using the forearm direction method. Thus, we use the head-finger line method. However, these conventional methods [4, 6, 13] do not take individual ways into account.

Some studies considered individual ways using regression models [3, 5]. These methods require a large amount of pointing gesture data with given objects for learning. Unlike the calibration for touch panel devices, in these methods, it is a time-consuming task for users to set up the learning situation with several known objects. We therefore propose a simple calibration method without any given object positions but instead use the position corresponding to the center of the camera.

3. PROPOSED METHOD

Our proposed method utilizes the user's pointing gesture to the center of the camera to adapt to the individual ways using Kinect skeletal data. During the pointing gesture, we extract the head and fingertip positions, and calculate the two offsets; horizontal offset o_h and vertical offset o_v. These two offsets are the distances between the user's fingertip position and the line of the head and the center of the camera. Our main contribution is that we do not need to collect a lot of learning data obtained from pointing gestures with several given object positions, and we do not require any extra screen or projector devices to indicate the object positions to the user during the learning steps. If the robot needs to obtain the user's calibration data, the robot can just say, "please point to me for calibration" to ask the user to point at the robot's camera. It is easy for users to detect the robot camera position and to point at the robot.

In our study, we use the Kinect camera as the robot camera. In addition, we use the Kinect coordinate as the world coordinate, the x-axis points to the left, the y-axis points upward, and the z-axis points forward. We set the Kinect horizontally, so the y-axis is parallel with the vertical direction. Before using our proposed calibration method, some Kinect skeletal data needs to be located to improve the accuracy of the pointing direction estimation. Then, we first locate the head and fingertip positions. Next, we extract the calibration data by using a pointing gesture. Finally, we apply them to the pointing gesture, allowing the robot to obtain the user pointing direction based on the head and calibrated fingertip line.

3.1. Preliminary arrangements

3.1.1. Estimation of the head position

Kinect sometimes fails to estimate the head skeletal position if the hand blocks part of the head region through a pointing gesture, or if the robot camera captures the user from a side view. The head skeletal position error is particularly large along the x and z coordinates. In these cases, we locate the new head position P_h aligned with a line from the spine to shoulder-center skeletal positions. Figure 2 shows the new head positions. In Fig. 2, the yellow dots indicate the original head skeletal positions obtained

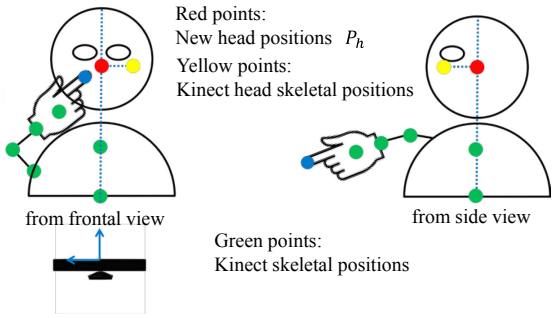


Fig. 2. New Localized Head Positions

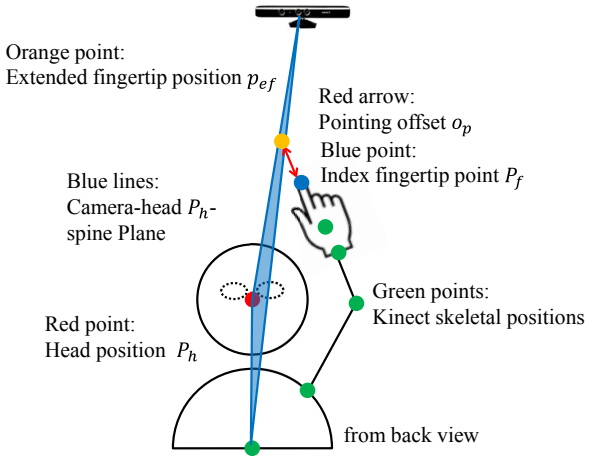


Fig. 3. Definition of the pointing offset

by Kinect. Then, we correct them with regard to the x and z coordinates using the spine and shoulder-center skeletal positions. If the spine and shoulder-center skeletal positions are blocked by a hand or a forearm, we use the original head skeletal position taken from Kinect as the new head position P_h.

3.1.2. Estimation of the fingertip position

Kinect can also extract human body regions as a point cloud, so with these data and skeletal data we extract fingertip position P_f. We simply define the fingertip as the extremity of the human point cloud region in the direction from the elbow to hand skeletal positions. Kinect sometimes fails to extract edge regions, so if there are no candidates for the fingertip positions we just use the hand skeletal position as the fingertip position. In most cases, fingertip position P_f is close to the actual fingertip position. Furthermore, we calibrate the fingertip position in the next step.

3.2. Calibration for the fingertip and angle

Using new head position P_h and fingertip position P_f, we calibrate the individual gesture way by means of a pointing gesture that points at the center of the camera. We measure the pointing offset o_p and the angular offset o_θ for every subject. We describe these procedures below.

3.2.1. Estimation of the pointing offset

Figure 3 shows the definition of pointing offset o_p. This pointing offset corresponds to horizontal offset o_h. As we mentioned above,

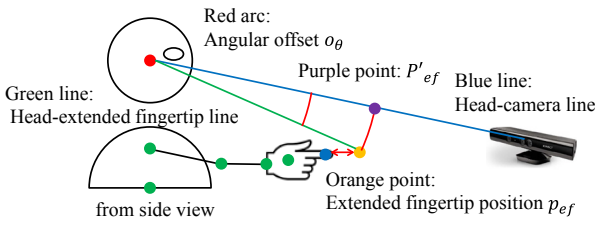


Fig. 4. Definition of the angular offset

it is difficult to extract the fingertip position. Therefore, we calculate an extended fingertip position to compensate for individual differences. The extended fingertip position P_{ef} is the intersection of the plane and the line; the plane is composed of the head position P_h, the spine, and the center of the camera. Furthermore, the line is parallel to the elbow-hand line from the fingertip position. During the action of pointing at the center of the camera, the elbow-hand line is almost perpendicular to the camera plane. In this case, Kinect sometimes fails to extract these arm positions precisely. Hence, we calculate the consistency of the positions among the hand, wrist, and elbow. During the pointing gesture, these three positions are usually on the same line. However, if these positions are inconsistent, we use the original fingertip position as the extended fingertip position. For example, the wrist position is at a specific distance from the hand-elbow line. In this case, we assign the pointing offset o_p to zero.

3.2.2. Estimation of the angular offset

After extracting the extended fingertip position, we calculate the angular offset o_θ. The angular offset o_θ is the angle between two lines; the line from the head to the center of the camera, and the line from the head to the extended fingertip position. Figure 4 shows how angular offset is defined. This angular offset corresponds to vertical offset o_v. If the user points at the object with just a small pointing gesture [9], this vertical offset is relatively large. Even in this case, our method can adapt to a small pointing gesture.

3.3. Estimation of the pointing direction

When we estimate the user pointing direction, we update the extended fingertip position by adding the pointing offset o_p from the fingertip position P_f in the direction of the elbow-hand line. Next, we rotate it by angular offset o_θ around the head position P_h vertically. Then the new extended fingertip position is p'_{ef} (in Fig. 4). Finally, we define the pointing direction as the line from P_h to P'_{ef}.

We can extract these two offsets for each frame in a pointing gesture to the center of the camera. Then we store the mean values of these two offsets respectively for every subject. These two means can also take a negative value. While we can calculate the mean values, we set threshold ranges for the pointing offset o_p and the angular offset o_θ for each frame. The absolute minimum of the pointing offset o_p is 0.05 m and the absolute minimum of the angular offset o_θ is one degree. If the value is under the threshold, we assign the value to zero. In this case, the pointing direction can be represented by the head and fingertip line because the user points to the object carefully. Meanwhile, we assign 0.25 m as the absolute maximum of the pointing offset o_p and ten degrees as the absolute maximum of the angular offset o_θ to prevent outliers.

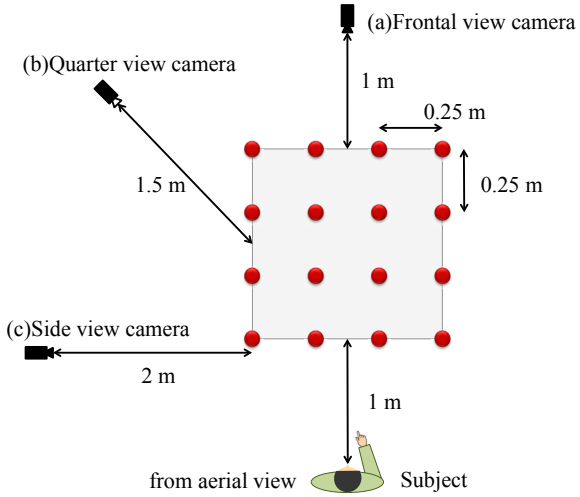
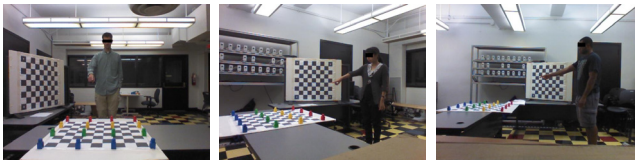


Fig. 5. Layout of our experiment



(a) Frontal View (b) Quarter View (c) Side View

Fig. 6. Examples of the test images

4. EXPERIMENTS AND RESULTS

4.1. Experimental environment

In order to evaluate the accuracy of the pointing direction, we conducted the experiments in an indoor environment. We assume a scenario in which the robot will fetch an object to which a user points. In addition, the test objects are relatively small (about a three-centimeter cube), and they are all on a table. Figure 5 shows the layout of our experiment from above. There are 16 objects arranged 4 by 4, and the objects are about 0.25 m apart. The table height is 0.7 m. We also checked the accuracy of the pointing direction with different camera angles in respect to body orientation. We set up cameras in three different locations: frontal, quarter, and side views. All camera height positions were 1 m. We used a Kinect camera, and then we asked the subjects to point at each of the objects three times. We measured the object 3d positions relative to the camera employing a manual operation using Kinect depth data.

We tested ten subjects, nine males and one female. Their heights ranged from 1.6 to 1.9 m. Nine subjects were right-handed and one was left-handed. Each calibration datum was captured once from the frontal view and all gestures were performed in the standing position. We collected data on a total of 480 pointing gestures in our experiment. Examples of the captured data are shown in Fig. 6.

We determined the target object by selecting the one closest to the line of the pointing direction. The scores ranged from zero to one with one being the best score. We compared the seven methods set out below to estimate the pointing direction.

(1) Kinect head and Kinect hand line-based method (K_hK_h)
(2) Kinect head and proposed fingertip line-based method (K_hP_f),
(3) Kinect dominant eye position and proposed fingertip line-based method (K_eP_f),
(4) Proposed method considering just the line from the head to the fingertip positions (P_hP_f),

(5) Proposed method considering the pointing offset (P_hP_{fo})
(6) Proposed method considering the pointing offset and the angular offset ($P_hP_{fo\theta}$)
(7) The regression method based on training with 16 objects (Train) [3]

Regarding method (3), the detection accuracy of the eye position is sometimes low from (b) Quarter view and (c) Side view. In these cases, we use the Kinect head position as the Kinect dominant eye position. Moreover, we used the subject's dominant eye for estimation. We split the data into two groups, one for training with 16 objects for method (7), and the other was used as test data to calculate the accuracy of all methods. Regarding training data, we segmented the pointing gesture frames manually. This is not ground-truth data, but it is close to the precision limit in our experiment.

4.2. Results

Figure 7 shows the average results for object identification accuracy using three different camera positions for methods (1) to (4), and figure 8 shows the average results for methods (4) to (7). The x-axis represents the methods arranged by the perspectives, the y-axis represents accuracy, and the error bar represents standard error (S.E.). In Fig. 7, it is clear that our proposed preliminary arrangements method (4) is able to improve the accuracy of object identification for all views, especially for the side view. Therefore, this finding suggests that we should use the located head position when the user points in a standing position. In Fig. 8, we can see that the proposed method has improved accuracy compared to method (4): the head-fingertip method. Furthermore, the accuracy of the proposed method is close to that of the regression method. From the perspective of the time-consuming task for learning, our method is useful for pointing estimation. Note that, from the comparison of the accuracy among (a), (b), and (c) camera views, the highest level of accuracy is from (b) quarter view, and the accuracy is as about the same as that as (c) side view and (a) frontal view. Many conventional methods [1, 2, 4, 6, 9, 10] captured the pointing gestures from a frontal camera. We assume that the easiest way is to detect the users from a frontal camera. However, Kinect is now sufficiently robust to detect humans from different camera angles. Therefore, if the robot can predict the moment when a user points, then moving to the quarter view relative to the user's body orientation will improve the accuracy of estimating the pointing direction.

Next, we discuss the experimental results in more detail. We assess the effect of the two offsets. To do this, we arranged four objects lengthwise and analyzed how we could discriminate the object groups widthwise. In the same way, we arranged four objects widthwise and analyzed how we could discriminate the object groups lengthwise. Table 1 shows the accuracy of the estimation for two groups. From this table, we can see that in terms of horizontal accuracy, the accuracy is relatively high for all methods. This means horizontal offset o_h is relatively smaller than the vertical offset o_v. We assume that it is because we look at the tip of the index finger from above during the pointing gesture. Furthermore, we can see that our proposed method improves accuracy for object groups widthwise.

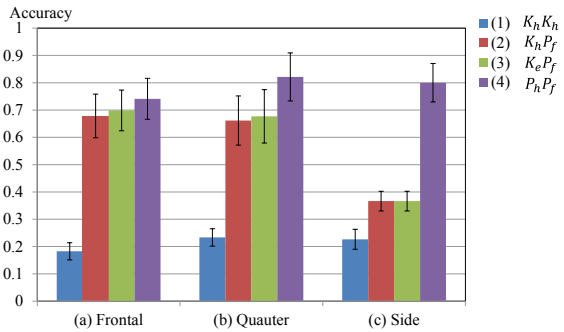


Fig. 7. The results of the conventional methods and the proposed head-fingertip method

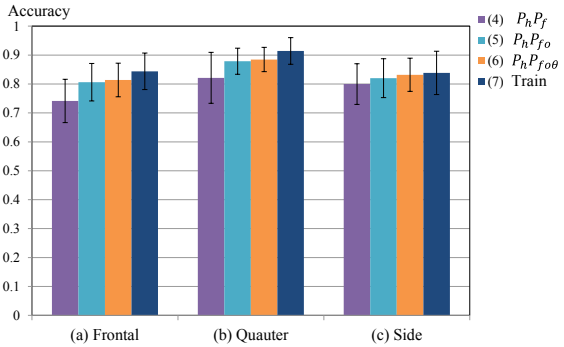


Fig. 8. The results of the proposed methods and the regression-based method

Table 1. The results from the viewpoints of the horizontal and vertical offsets

Camera Angles	Methods	Horizontal	Vertical
(a) Frontal	(1)	0.49	0.24
	(2)	0.90	0.69
	(3)	0.91	0.64
	(4)	0.91	0.75
	(5)	0.96	0.81
	(6)	0.98	0.82
	(7)	0.98	0.85
(b) Quarter	(1)	0.80	0.24
	(2)	0.90	0.71
	(3)	0.83	0.67
	(4)	0.95	0.83
	(5)	0.99	0.88
	(6)	0.98	0.90
	(7)	0.98	0.93
(c) Side	(1)	0.80	0.24
	(2)	0.86	0.40
	(3)	0.86	0.40
	(4)	0.93	0.83
	(5)	0.93	0.87
	(6)	0.93	0.89
	(7)	0.98	0.86

5. CONCLUSION

In this paper, we proposed a method for estimating users' pointing direction by means of efficient calibration. The proposed method uses the head position and the calibrated fingertip position to estimate the pointing direction. We calculate the calibrated fingertip position while the user points to the center of a camera. Our method outperforms other conventional methods; head-hand, head-finger, and eye-finger line-based methods.

We assume that the offsets depend on the relative positions among the user, the object, and the camera. In the future, this study will be extended to encompass many different positions in an indoor situation.

6. REFERENCES

[1] M. V. Bergh, D. Carton, R.D. Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. V. Gool and M. Buss, "Real-time 3D Hand Gesture Interaction with a Robot for Understanding Directions from Humans," *20th IEEE International Symposium on Robot and Human Interactive Communication*, pp.357-362, 2011.

[2] B. Burger, I. Ferrane, F. Lerasle, and G. Infantes, "Two-handed gesture recognition and fusion with speech to command a robot," *Autonomous Robots*, Vol. 32, Issue 2, pp.129-147, 2012.

[3] D. Droeschel, J. Stuckler, and S. Behnke, "Leaning to Interpret Pointing Gestures with a Time-of-Flight Camera," *sixth ACM/IEEE International Conference on Human-Robot Interaction (HRI),* pp.481-488, 2011.

[4] P. Jing and G.Y. Peng, "Human-computer Interaction using Pointing Gesture based on an Adaptive Virtual Touch Screen," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, Vol.6, No.4, p. 81-92, 2013.

[5] N. Jojic, B. Brumitt, B. Meyers, S. Harris, and T. Huang, "Detection and estimation of pointing gestures in dense disparity maps," *Fourth IEEE international Conference on Automatic Face and Gesture Recognition*, pp.468-475, 2000.

[6] Z. Li and R. Jarvis, "Visual Interpretation of Natural Pointing Gestures in 3D Space for Human-Robot Interaction," *Control Automation Robotics & Vision (ICARCV)*, pp.2513-2518, 2010.

[7] C. Martin, F-F. Steege, and H-M. Gross, "Estimation of Pointing Poses for Visual Instructing Mobile Robots under Real World Conditions," *Robotics and Autonomous Systems*, Vol.58 Issue 2, pp. 174-185, 2010.

[8] K. Nickel and R. Stiefelhagen, "Visual recognition of pointing gestures for human-robot interaction," *Image and Vision Computing*, Vol.25, Issue 12, pp.1875–1884, 2007.

[9] C.-B. Park, and S.-W. Lee, "Real-time 3D pointing gesture recognition for mobile robots with cascade HMM and particle filter," *Image and Vision Computing*, Vol.29, Issue 1, pp.51–63, 2011.

[10] M. Sigalas, H. Baltzakis, and P. Trahanias, "Gesture recognition based on arm tracking for human-robot interaction," *Intelligent Robots and Systems (IROS),* pp.5424-5429, 2010.

[11] R. Steifelhagen, H. K. Ekenal, C. Fugen, P. Gieselmann, H. Holzapfel, F. Kraft, K. Nickel, M. Voit, and A. Waibel, "Enabling Multimodal Human–Robot Interaction for the Karlsruhe Humanoid Robot," *IEEE Transactions on Robotics*, Vol. 23, Issue 5, pp. 840 - 851, 2007.

[12] S. Waldherr, R. Romero, and S. Thrun, "A Gesture Based Interface for Human-Robot Interaction," *Autonomous Robots,* Vol.9 No. 2, pp.151-173, 2000.

[13] Y. Yamamoto, I. Yoda, and K. Sakaue, "Arm-pointing gesture interface using surrounded stereo cameras system," *17th International Conference on Pattern Recognition (ICPR)* , Vol. 4, pp.965-970, 2004.

[14] Kinect for Windows SDK v1.7