Perception and Control Challenges for Effective Human-Robot Handoffs

Vincenzo Micelli* Kyle Strabala† Siddhartha S. Srinivasa†
*Department of Information Engineering, University of Parma, {micelli}@ce.unipr.it
†The Robotics Institute, Carnegie Mellon University, {strabala, siddh}@cmu.edu

***Abstract*—Human-robot handoffs are a primary task for a personal robot. Previously, robots put the majority of the handoff burden on the human, where in the case of the human to robot handoff the human had to position and orient the object into the robot's hand. This works well because it is a very easy task for humans, but what happens if humans handoff objects while they are distracted or if they have some impairment? Is the handoff still possible? The recent arrival of reliable human-tracking systems like the Kinect and smart control algorithms that use the human-tracking feedback and other sensor data can allow for rich human-robot interaction where the robot is aware of the human and can actively collaborate together towards a final goal. In this paper, we use rich 3D RGB-D data for human-tracking and object detection, combined with incremental control methods, to create a handoff system that is reliable and allows the robot to actively collaborate in a handoff with a human.**

I. Introduction

Handoffs are an important function in everyday life. For humans, handoffs are an easy task and are usually routine rather than deliberative. Even if a human performs several handoffs on a daily basis with different types of objects and in different situations, they cannot remember how exactly they performed the handoff. Also, the motion of the giver and the receiver is often synchronized and they work together until the transfer of the object is complete [1].

What happens when one of the humans is replaced by a robot? Edsinger at al [2] demonstrate that subjects without explicit instructions can successfully hand objects to a robot and take objects from a robot in response to reaching gestures. Moreover, when handing an object to the robot, subjects usually fix the position and the pose of the object to match the configuration of the robot's hand. Finally, the robot knows to open and close his hand when it senses a force on its end-effector, meaning the human must physically push or pull the object into the robot's stationary hand for the handoff to be successful.

Other recent results include the AIST HRP-2 [3] and the HERMES robot [4]. While these publications include handoffs between a robot and a human, they are not focused on the performance of the handoff nor the burden on the human to complete the handoff.

In the works above, robots put the majority of the handoff burden on the human. In the case of the human to robot handoff, the human has to position and orient the object into the robot's hand and then push for the robot to know that it should close its hand. This works well because it is an easy task for humans, but if humans handoff objects while they are distracted the robot needs to participate actively to complete the handoff. We could not find any previous work on handoffs where the robot takes an object instead of receives it. The main reason that this has not been done is that it is very hard to reliably track humans.

The recent arrival of sensors like the Microsoft Kinect can enable robotic systems to perceive the presence of humans. The Kinect is a RGB-D camera that records two images, a RGB image and a depth image. These images can be used by the software bundled with the Kinect to detect and track humans in a fast and reliable way. This new technology can enable the active participation of robots in human-robot handoffs that was previously unavailable by improving the perception capabilities of robots. However, the Kinect human tracking is not perfect and is sensitive to fast human movement, non-frontal views of humans, and situations where the human is near other objects.

In this paper we present our progress towards a working human to robot handoff algorithm, where the robot actively takes the object from the human. In Section III, we talk about how we sense what the human is doing including compensating for noise from the human tracking software, detecting handoff intent, and detecting objects in the human's hand. Then, in Section IV we talk about two different robot control methods. In Section V we outline an informal study that we used to test our algorithm. Finally in Sections VI and VII we talk about the results of our user-study and what we learned from this work about human-robot handoffs.

II. The Framework

We use HERB, the Home-Exploring Robotic Butler [5]. HERB has two Barrett WAM arms mounted on a segway base that enable him to move around an environment and perform advanced manipulation tasks. HERB has a suite of sensors to help him perceive the world, including a spinning laser scanner for building 3D world models, a vision system for object recognition and pose estimation [6], and a commercial system for indoor localization.

HERB typically works in a domestic kitchen environment, as shown in Fig. 1. In this paper, HERB is positioned next to and facing a table which has two types of items on it: Pop-Tarts boxes and Fuze bottles. We have already demonstrated

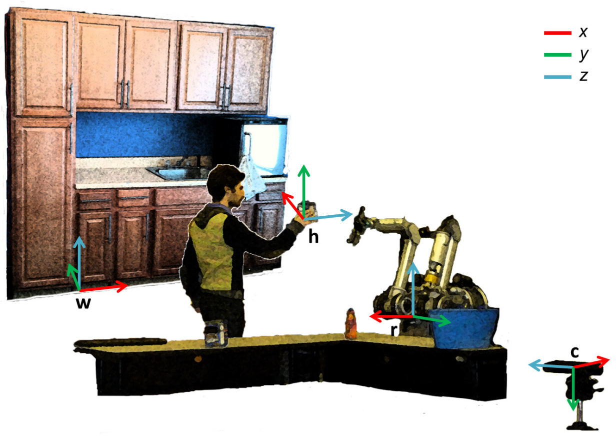


Fig. 1. Scenario.

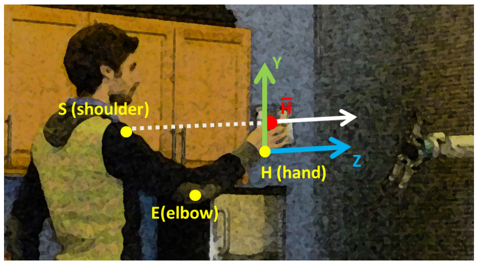


Fig. 2. Hand frame.

HERB's ability to recognize and manipulate these objects autonomously [5], while avoiding humans safely. In this paper, our goal is to explore how HERB can actively collaborate with a human to accomplish the task.

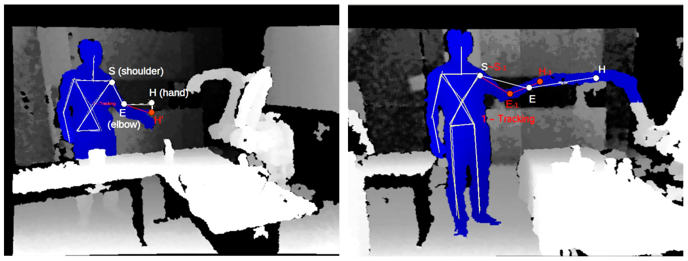
A Microsoft Kinect camera is located $2.5m$ from the table in order to have a complete view of the scene. Images from the camera provide two types of data at $30Hz$: raw depth and color, and human tracking data. The human tracking data is a 14-point skeleton outlining a human's pose from head to feet.

We combine information from four different coordinate frames (Fig. 1): the fixed world frame w located at the bottom of the kitchen cabinets, the fixed camera frame c located on the Kinect camera, the robot frame r located on HERB's base, and the moving human hand frame h located on the hand.

We use the HERB's localization system to obtain its transform r. We calibrate the extrinsics of the camera c with a calibration procedure where salient point correspondences on the kitchen cabinets and on HERB are matched. When we detect a human, we compute a hand frame h online (Fig. 2). We use this frame as a target for the robot's planning and control. The Z-axis of the hand frame lies on the world X-Y plane and points along the shoulder-hand direction, and the Y-axis points upwards opposing gravity.

III. Perception

Although the Kinect provides useful human tracking data, we had to address various perception issues to make it suitable



(a) White lines: OpenNI skeleton tracker output. Red line:corrected hand position (b) Robot and human arm point clouds have merged.

Fig. 3. Skeleton tracking

for handoffs. Our post-processing included refining the skeleton tracking for better wrist positions particularly when the human is close to or partially occluded by HERB, detecting the intention of a human to handoff an object, learning to identify if an object was held and recognizing the object type, and calculating where HERB should put his hand for the handoff.

A. Correcting Hand Pose

Although the Kinect human tracking system is reliable, the tracked position of the hand has been observed to have errors. These errors occur when the human is moving fast and when the human is near or touching other objects.

In the former case, the human tracker often lags the true human motion. We address this issue with a correction that moves the hand to the closest 3D point in a small box around the estimate (Fig 3a) . This correction is fast, and is naturally robust: if the hand tracking is originally good, the correction does not change the estimate.

In the latter case, when HERB is close to the human, the two are often partially merged into the human point cloud. This is of particular concern to us since the human and the robot need to be close to each other in the final part of the handoff (Fig 3b). We take advantage of the fact that we can query the joint configuration of the robot. We filter out data when the robot hand and the human hand are closer than $15cm$ and the robot hand position is embedded within the human point cloud. Since the human does not move much when very close to the robot, this approximation is far better than incorrect tracking.

B. Handoff Gesture

After correcting the hand pose, we infer impending handoff based on two features: 1) the human is in a handoff pose and 2) the human is holding an object. We detect the former by analyzing the human skeleton, and the latter using a support vector machine (SVM) to classify an image patch around the detected hand.

1) Handoff Pose Detection: Humans use several cues to signal a handoff including speech, gaze, motion, and posture. In this paper, we focus on postural cues inferred from human tracking. We used the following cues, motivated by studying human-human handoffs, to infer the handoff signal:

1) The human is near HERB.

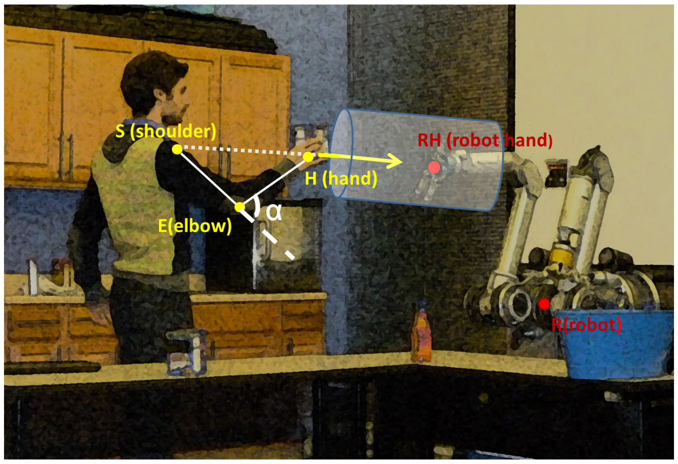


Fig. 4. Handoff pose detection.

2) The vector from the shoulder to the hand points towards HERB's hand or upper body, as shown in Fig. 4.
3) The human elbow is bent at an angle $\alpha \leq \alpha_{\max} \equiv 150^o$.

To avoid false positives when the human is moving and momentarily assumes a handoff pose, we trigger a detection only if the human is in the handoff pose for 5 consecutive frames (which at 5fps is $1 sec$).

2) Object detection: People often wave their arms and assume postures that are akin to handoffs. To reliably detect a true handoff, we found it critical to detect if the human was actually holding an object before starting a handoff response in HERB. To enable this, we developed an efficient and reliable object detection system that detects if the human is holding an object, and identifies the held object.

Our system is composed of two main components: an algorithm for extracting the bounding box around the human hand, and a support vector machine (SVM) [7] that classifies the bounding box.

The algorithm for extracting the bounding box consists in the following steps (Fig. 5):

1) Obtain a depth image in the camera frame c (Fig. 5a).
2) Transform the depth image into the hand frame h. Crop to an axis-aligned bounding box (Fig. 5b).
3) Decimate the depth into clusters of contiguous points, removing stray outliers. The hand is often detected as a single cluster (Fig 5c). However, sometimes the hand and the robot are detected together as one cluster (Fig 5d).
4) Since the transformed depth image is centered at the hand, the cluster closest to the origin is labeled the hand cluster.
5) To detect if HERB's hand is clustered with the human hand, we use forward kinematics to determine the pose of HERB's hand and check if it lies within the cluster (Fig 5e).
    a) If HERB's hand is within the hand cluster, bypass object detection and trigger the final phase of handoff.
    b) If not, we crop the human hand. We assume that the object is held farthest from the hand and fit a smaller box tightly around the points. (Fig. 5f). If the human is not holding anything, the smaller box contains just the hand.

Features	Training Set	Test Set	Cross Validation	Prediction Accuracy
RGB (48)	4766	1623	99.96%	93.16%
RGB (48) + Height (1)	4827	1623	99.96%	**96.61%**

TABLE I: SVM PERFORMANCE.

Once the final box is computed, we compute color and geometry features and learn an SVM [7] to predict three classes: Pop-Tart boxes, Fuze bottles, and empty hands. SVMs have proven to be quite useful for data classification We use color histograms (16 bins for RGB, with a total of 48 features) and the height of the bounding box (1 feature) and an RBF kernel.

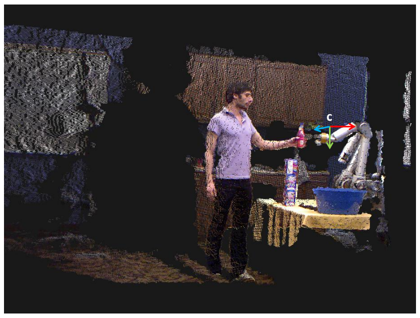
Despite their simplicity, color histograms have demonstrated good results in practice [8], [9]. We found that although the color features are often able to predict the object correctly, the height feature allows us to distinguish better between objects with similar color but different height, like for instance a Fuze bottle and a pink empty hand. Table I reports the results achieved with our algorithm considering the same testing data with and without the height feature.

Because the object is small and far from the camera, we found local descriptors like NARF [10] to be far less useful when compared with global descriptors like color histograms. For the same reason, we found the depth information in the bounding box to be far too corrupted by noise and quantization to be useful beyond a global descriptor like the height of the object.

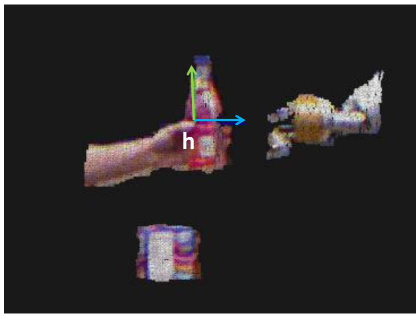
IV. Planning and Control

As soon as the perception system triggers an impending handoff, HERB moves to take the object. We compared two autonomous motion strategies, a planner and a controller. Both strategies receive the same input: a target human hand pose from the perception system at 5fps. Both strategies control the arm via joint velocities (converted internally by the arm driver into joint torques) and have access to the 6 axis force-torque sensor on the wrist for detecting forces.

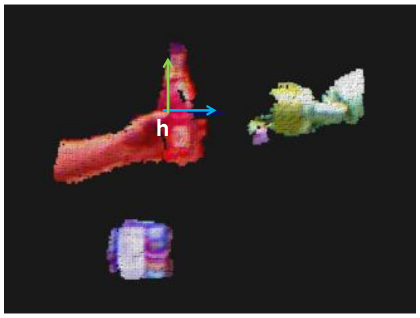
The execution of these two strategies differs in the temporal position of the planning phase with respect to the execution loop as shown in Figure 6. The controller plans in the near future during each iteration of the execution loop, while the planner plans for the entire execution before the execution loop begins. The effects of this are that the controller begins execution immediately but cannot predict problems in the future like joint limits and collisions, while the planner takes several (≈ 5) seconds to plan the entire trajectory that avoids joint limits and collisions. The details of each planner are presented in the following two sections.



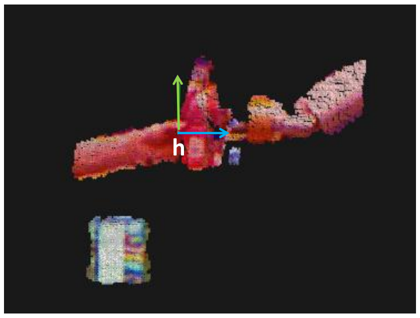
(a) *The scene.*



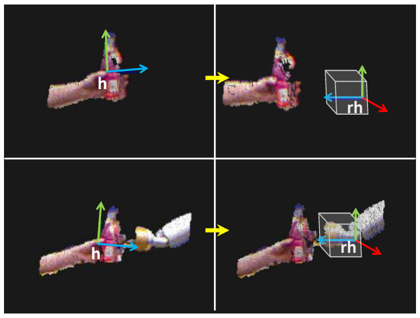
(b) *Bounding box around the transformed hand frame.*



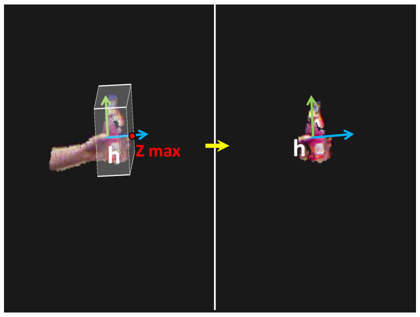
(c) *Clusters where the human's hand (red) and HERB's hand are separated.*



(d) *Clusters where the human's hand (red) and HERB's hand are not separated.*



(e) *Extracting the hand cluster.*



(f) *Refining the bounding box.*

Fig. 5. Stages of the bounding box extraction algorithm.

A. Planner

The planner takes the very first reported human hand pose and plans to get to it. We use a randomized planner that we have developed that efficiently explores high-dimensional constraint manifolds [11]. The planner is tasked with producing a feasible, collision-free, reasonably smooth path as quickly as possible. Any updates to the human hand pose are ignored during planning. As soon as a path is returned, the planner executes it.

In our case, the planner takes as input a starting arm configuration and a goal end-effector pose, consisting of the end-effector position and orientation. With the goal pose, the planner finds several candidate goal IK solutions. Then, the planner expands random trees from each of the start and candidate goal configurations until a path is found from start to goal that is collision free and within the joint limits. This path is often very jagged, so the final step is to run the path through a trajectory smoother to remove backtracking and corners.

Randomized planners are probabilistically complete: guaranteed to find a feasible path if one exists. However, more search time is required in situations with more constrained configuration spaces. Also, while the output of the planner is smoothed, there is no guarantee of global or local optimality.

B. Controller

The controller is an implementation of inverse Jacobian control with constraints [12]. The Jacobian relates angular velocities of the arm joints to hand velocities. For the controller,

```
CONTROLLER( )
    while CLOSEHAND( ) == False do
        $\theta \leftarrow$ GETCURRENTJOINTANGLES()
        $goal \leftarrow$ GETCURRENTGOAL()
        $d\theta \leftarrow$ CALCULATEJOINTINCREMENT($\theta, goal$)
        GOTOCONFIG($\theta + d\theta$)
PLANNER( )
    $\theta \leftarrow$ GETCURRENTJOINTANGLES()
    $goal \leftarrow$ GETCURRENTGOAL()
    $path \leftarrow$ CALCULATEPATH($\theta, goal$)
    while CLOSEHAND( ) == False do
        $t =$ GETCURRENTTIME()
        GOTOCONFIG($path(t)$)
```

Fig. 6. Controller vs Planner Algorithms.

we use two Jacobian operators, one relating joint angular velocities to end-effector velocity, J_X, and the second relating joint angular velocities to end-effector twist via the quaternion velocities, J_q. Combining these two Jacobian into one gives J as

$$J = \begin{bmatrix} J_X \\ J_q \end{bmatrix}$$

The Moore-Penrose pseudo-inverse of J is

$$J^+ = \text{PseudoInverse}(J)$$

Then, for a given change in the end-effector position δX and orientation δq, the approximate change in joint angles required to accomplish this change, $\delta\theta_{pose}$, can be calculated as

$$\delta\theta_{pose} = J^{+} \begin{bmatrix} \delta X \\ \delta q \end{bmatrix}$$

HERB has a redundant DOF meaning that he has more arm freedom than is required to achieve a 6 DOF pose of his end-effector. HERB has 7 joints to position the end-effector in 6 DOF, so in the case where his arm is not in a singularity, HERB can move about in the null-space of his Jacobian without changing his end-effector pose. We can move around in the null space to accomplish several different goals. For instance, avoiding joint limits, avoiding singularities, minimizing joint velocity, and minimizing joint torques. For this controller, the null space has 1 DOF and is used to avoid joint limits. The change in joint angles to minimize the difference between the current joint angles θ and desired joint angles θ_{des} is

$$\delta\theta_{limits} = N_J N_J^T \cdot (\theta - \theta_{des})$$

where N_J is the null space of the Jacobian.

Finally, the output joint angular velocity $\dot{\theta}$ is a combination of the two joint differentials with some scaling values α and β.

$$\dot{\theta} = \alpha \cdot \delta\theta_{pose} + \beta \cdot \delta\theta_{limits}$$

Up to this point the controller has no concept of collisions or exceeding joint limits. Therefore, before commanding the desired joint velocities, collision checks and joint limit checks must be performed. If there is a collision or joint limit problem, then the joint velocities are commanded to zero and the controller is stuck until the desired joint velocities produce a path that is collision free and within the joint limits. To force movement in these situations, a planner like that in the last section can be used to move away from the problem.

C. Completing the handoff

Both strategies use the same method for detecting when to close the hand to complete the handoff. There are two ways to initiate a close hand command, force into the robot's hand and a timeout. We found that both triggers are required for a reliable handoff. If only the force trigger is used, then the robot sometimes hovers around the object and the users do not intuitively know push the object into the robot's hand. On the other hand, if only the timeout is used, then the robot will not respond to contact with the human which makes the robot seem very aggressive to the user. The force is calculated using a 6 axis force-torque sensor located at the wrist of the arm. If the force into the palm is greater than $5N$, then the arm stops moving and a close hand command is initiated. The force-torque sensor can sense forces smaller than $5N$, but the mass of the hand creates a non-zero measurement when the arm moves. To trigger the close hand command with the timeout, the robot hand must be stationary at the given human hand location for at least 0.5 second. If the hand is closed due to force or the timeout, then success is returned based on whether or not the fingers close all the way into a fist. If the hand closes into a fist, then the handoff is reported as failed, otherwise, the handoff is reported a success.

V. User Study

We wanted to test HERB's ability to take objects from a human during a handoff while minimizing the effort contributed by the human. To this end, we created a user study where the subjects were told to perform two tasks at once, a computer task and a handoff. The subjects were told to focus their attention on the computer task, thereby letting HERB take the object from the human while getting little or no help from the subject.

The user study consisted of 5 subjects, tested individually. The subjects were seated at the table with a monitor, mouse, and 7 objects in front of them. HERB was positioned to the subject's left where he could be seen in the subject's peripheral vision and visible if the subject looked away from the monitor. The subjects were instructed to play a computer task and, when prompted by HERB, to handoff one of the objects to HERB. The subjects used their right hand to move the mouse and used their left hand to perform handoffs to HERB (Fig. 7).

Two control algorithms were tested in sequence. The subjects performed 7 handoffs with the controller, had a short break of 30 seconds, then performed 7 handoffs with the planner. Failed handoffs were recorded and if the subject still had the object, the object was taken away by an investigator.

The computer task was a slightly modified version of the PEBL Continuous Performance Task. The computer task presented the subject with a small target on the monitor where the subject was to move the mouse. Once the target was reached, the subject was presented with another target in another location. In the original computer task, the subject clicked in between targets to signal his readiness. In our version there is no delay between targets. To make the computer task more challenging, the mouse input was summed with a random error which made the mouse quiver by a small amount. This addition of noise required more attention from the subject, leaving less attention for the handoff.

After completing the handoffs with both control algorithms, each subject was asked to compare the two controllers as well as state any comments they had. Subjects were asked to choose which controller they preferred in 5 areas and state why. The 5 areas were preference, natural-looking, easier, safer, and human-like.

VI. Results

We had 5 subjects participate in our informal study. Overall, the combination of perception with the Kinect data and a controller with a take attribute resulted in a handoff success rate of 83% for the 70 handoff attempts. The majority of these attempts were with the users completely distracted, just holding the object up and waiting for the robot to take it while they continued with the computer task.

The two control algorithms were compared by number of successes and total handoff time. The controller was faster than the planner by a factor of nearly 2, taking an average time of

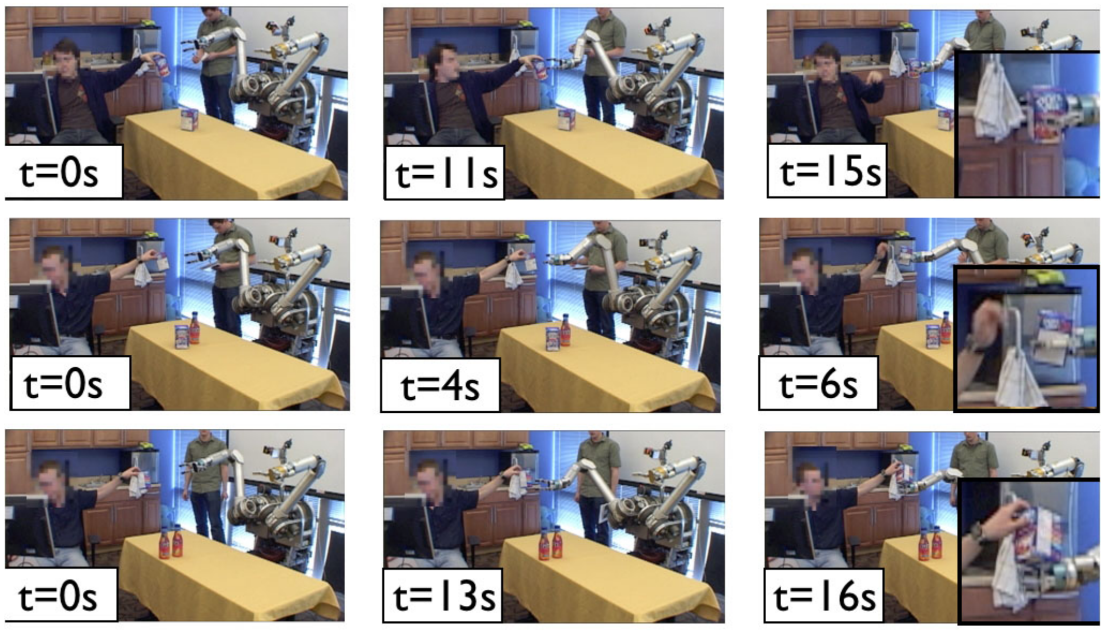


Fig. 7. Results: Top: Planner success, Middle: Controller success, Bottom: Planner failure

8.46 seconds from detection to the grasp finishing while the planner took an average time of 14.23 seconds. These handoff times include the time it takes to close the hand, usually around 2 seconds. The two controllers are similar in timing, except the planner has an additional $4sec$ planning phase tacked on at the beginning. The paths that the two controllers executed were sometimes quite different. The controller always takes a predictable curved path to the goal, whereas the planner can have different unpredictable trajectories even with the same input, as shown in Figure 8. For the human, working with an unpredictable robot can be disconcerting.

The planner had more successes (91.43%) than the controller (74.29%) as shown in Fig. 9. Failures with the controller were caused by three problems: the controller hit the arm joint limits and became stuck, the object detector failed to detect the object for the duration of the handoff attempt which caused the robot hand to retreat unexpectedly, and the object fell during the transfer. Failures with the planner were only caused by the object falling during the transfer since the planner avoids joint limits during its planning phase and only the first handoff detection is used to trigger the handoff attempt. If we exclude the handoffs with the controller that failed due to erroneous SVM object detection or joint limits, then the handoff success rate for the controller jumps to 90%, similar to the planner. The object detection algorithm recognized correctly 94% of the grasped objects.

After the test, subjects were asked about what they liked or disliked in the two control algorithms. Subjects found the planner aggressive while they liked that the controller was more ”gentle”. This happened because the velocity of the controller is directly proportional to the distance from the object. Moreover sometimes the planner executed strange trajectories that made users feel less safe since they didn’t understand what the robot was doing.

The subjects noted in both the algorithms characteristics like:

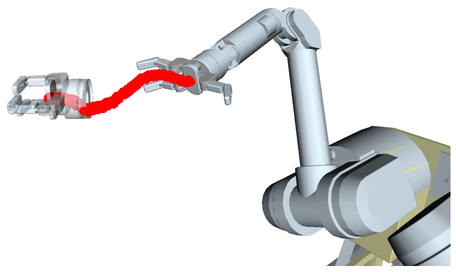
- Forcefulness: some of the subjects liked pre-grasp touching since it provided feedback of what was happening, others didn’t like that the robot pushed against their hand.
- Aggressiveness: subjects felt the robot was being aggressive when it approached the object quickly.
- Predictability: subjects liked it when they could predict what the robot was going to do, and, similarly, they became uncomfortable if the behavior of the robot was not predictable.
- Timing: subjects pointed out that human-human handoffs are faster.

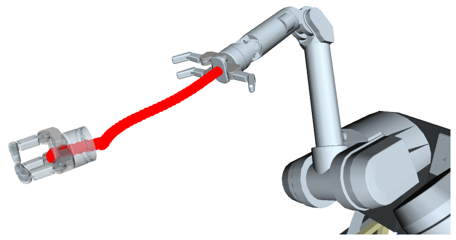
These parameters are independent of and tunable for both of the control algorithms.

The posture of the subjects during the handoffs varied from relaxed to fully extended arms. The relaxed posture forced HERB to go and take the object, whereas the fully extended arms positioned the handoff object as close to the robot’s hand as possible. Often a subject would start either fully extended or relaxed and then gradually switch as the handoffs went on. The subjects rarely moved the object once stopped, meaning that the controller and planner would go to the same place. If the subjects had moved after a handoff was detected, only the controller would have successfully followed the motion.

VII. Discussion

Previously, robots have not actively participated in handoffs. The robot controls whether or not a handoff occurs merely by either presenting his hand or not. Once the robot shows


(a) Typical controller trajectory


(b) Good planner trajectory

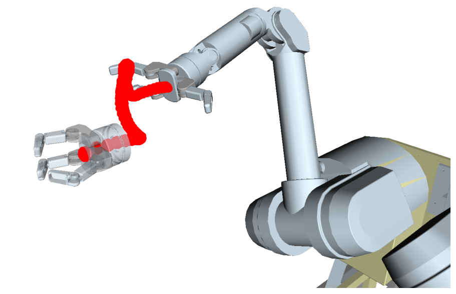

(c) Bad planner trajectory

Fig. 8. Hand trajectories for three handoffs, one with the controller and two with the planner. The bad planner trajectory was chosen to illustrate the random nature of the planned paths.

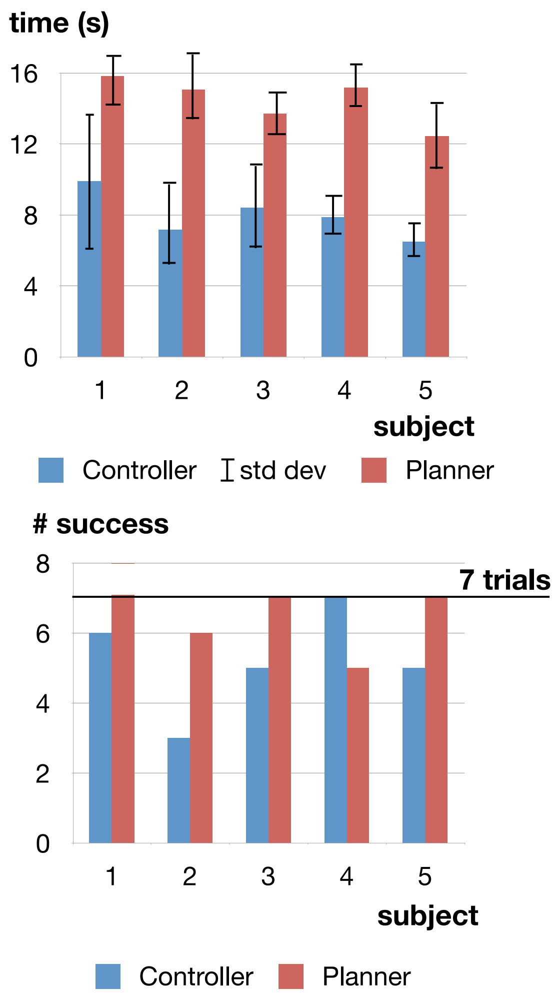


Fig. 9. User study results: (Top) Average time to handoff, (Bottom) Number of successful handoffs

that he is ready for a handoff, the entire handoff burden is put on the human, making the human position and orient the object into the robot's hand and then signaling the robot that he should close his hand either by force-feedback or other means. When the human is focused on performing the handoff, this handoff method works well because it is very easy for humans to put the object in the hand of the robot. But if the human is distracted with something else and cannot focus on the handoff, he needs the robot to help with the handoff.

In this paper we have presented an approach that allows HERB to actively participate in human-robot handoffs. This approach uses human tracking and 3D point data from the Kinect sensor to determine if a handoff is desired by a human and to control the arm to take the object from the human. With a user study we have shown that HERB is able to complete a handoff with a human even when the human is distracted by something else forcing HERB to actively take the object instead of receive it. Also, we have shown that HERB and a human can effectively and intuitively work together to complete the handoff without specific instruction to the human.

We used a point cloud of the scene to detect the object and compute the handoff point. Our algorithm extracts from the scene an image of the object without the background, which helps the object detection SVM and hand off point calculation. Using SVM's to detect objects in the hand gives us information for the handoff but also allows us to then use this object information in our behavior engine to decide program flow.

We have tested two different control algorithms. The controller beats the planner in total handoff time by $5sec$, but the planner beats the controller in success rate by 10%. While the two controllers each have their strong and weak points, we believe that the controller has the best potential. The controller is faster and can be made even faster with more aggressive gains. The controller is predictable in that it always heads straight for the goal, whereas the planner can be random in its motion. Predictability makes the human more comfortable around HERB because they can plan into the future and know that HERB won't do anything strange. Also, the controller is able to monitor the handoff point and update its target position in real-time. This means that HERB can follow the motion of the human during a handoff. While we did not see any motion of the object during the handoffs in the user study, we believe this capability could be important in other situations. The downside to the controller is that it can get stuck, but we believe these situations are rare and can be handled with a smart controller that can use a planned trajectory to get unstuck. In addiction, the controller was more subject to SVM failures because it checked the presence of the object for the duration of the handoff, so a SVM failure during the handoff made the robot retreat unexpectedly. We believe this type of failure can be removed in the future with some filtering of the SVM output.

In terms of future work, there are many possibilities. It would be interesting to test different types of triggers for the grasping, ie. is it better to have force and timeout triggers or to just immediately grasp the object when we reach the point, or use another type of sensor? Another interesting future work would be to use a second RGB-D camera on board HERB to enrich the point cloud. The camera on board would provide a more dense point cloud in the proximity of the object to use techniques like NARF [10] for object recognition. These kinds of techniques are robust to light variations and allow direct identification of an object instead of casting the object into a certain number of trained classes. In this case where we have two Kinect cameras, the first camera would give us information about human motion in the scene and the second on board camera would give us better information around of the points of interest near HERB.

The availability of reliable human-tracking systems like the Kinect and smart control algorithms that use the human-tracking feedback and other sensor data can allow for rich human-robot interaction where the robot is aware of the human and can actively collaborate together towards a final goal. In this paper, the use of rich 3D RGB-D data for human-tracking and object detection, combined with incremental control methods, created a handoff system that is reliable and can take some of the mental and physical burden off of the human. Continued research in this area will lead to very intuitive, comfortable and efficient human-robot interaction on a level not seen before in previous work.

Acknowledgment

This material is based upon work partially supported by the National Science Foundation under Grant No. EEC-0540865. Vincenzo Micelli is partially supported by EU project VAALID (FP7, Accessibility and usability validation framework for AAL interaction design process). Special thanks to members of the Personal Robotics Lab for their insightful comments and discussions.

References

[1] S. Shibata, K. Tanaka, and A. Shimizu, "Experimental analysis of handing over," in *4th IEEE International Workshop on Robot and Human Communication, 1995. RO-MAN'95 TOKYO, Proceedings.*, 1995.

[2] A. Edsinger and C. Kemp, "Human-robot interaction for cooperative manipulation: Handing objects to one another," in *The 16th IEEE International Symposium on Robot and Human interactive Communication, 2007. RO-MAN 2007.*, 2007.

[3] N. Sian, K. Yoki, Y. Kawai, and K. Muruyama, "Operating humanoid robots in human environments." in *In Proceedings of the Robotics, Science & Systems Workshop on Manipulation for Human Environments*, Philedelphia, Pennsylvania, August 2006.

[4] R. Bischoff and V. Graefe, "Design principles for dependable robotics assistants." *International Journal of Humanoid Robotics*, vol. 1, no. 1, 2005.

[5] S. Srinivasa, D. Ferguson, C. Helfrich, D. Berenson, A. Collet Romea, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. Vandeweghe, "HERB: a home exploring robotic butler," *Autonomous Robots*, vol. 28, no. 1, pp. 5–20, January 2010.

[6] A. Collet Romea, M. Martinez, and S. S. Srinivasa, "The MOPED framework: Object recognition and pose estimation for manipulation," *International Journal of Robotics Research*, 2011.

[7] C. Chang and C. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[8] O. Chapelle, P. Haffner, and V. Vapnik, "Support vector machines for histogram-based image classification." *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, 1999.

[9] M. J. Swain and D. H. Ballard, "Indexing via color histograms," in *Computer Vision, 1990. Proceedings, Third International Conference on.*, 1995.

[10] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard, "NARF: 3D range image features for object recognition," in *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, 2010.

[11] D. Berenson, S. S. Srinivasa, and J. Kuffner, "Task Space Regions: A framework for pose-constrained manipulation planning," *International Journal of Robotics Research*, 2011.

[12] O. Khatib, "The operational space formulation in the analysis, design, and control of manipulators," in *Proc. of the International Symposium on Robotics Research*, dec 1986.