Towards Perceptual Shared Autonomy for Robotic Mobile Manipulation

Benjamin Pitzer, Michael Styer, Christian Bersch, Charles DuHadway, Jan Becker

***Abstract*— Reliability and availability are major concerns for autonomous systems. A personal robot has to solve complex tasks, such as loading a dishwasher or folding laundry, which are very difficult to automate robustly. In order for a robot to perform better in those applications, it needs to be capable of accepting help from a human operator.**

Shared autonomy is a system model based on human–robot dialogue. This work aims at bridging the gap between full human control and full autonomy for tasks in the domain of personal robotics. One of the hardest problems for personal robotic systems is perception: perceiving and inferring about objects in the robot's environment. We present a system capable of solving the perceptual inference in combination with a human, such that a human operator functions as a resource for the robot and helps to compensate for limitations of autonomy.

In this paper, we show how a human-robot team can work together effectively to solve complex perception tasks. We present a system that asks a human operator to identify objects it doesn't recognize or find. In various experiments with the PR2 robot we show that this shared autonomy system performs more robustly than the robot system alone and that it is capable of tasks which are difficult to accomplish by an autonomous agent.

I. Introduction

Many scientists have pointed out the clear benefits of robots and humans working as partners [1], [2], [3]. The research fields of Adjustable Autonomy (AA) and Mixed Initiative Control (MIC) aim at bridging the gap between full human control and full autonomy. In many AA and MIC systems, the human operator is in charge of the main operation and autonomy is gradually added to support the execution of the operator's intent. For example, new telepresence robots support remote navigation with automatic obstacle avoidance [4]. The local autonomy (obstacle avoidance) of the robot decreases the cognitive load of the human operator and increases the effectiveness of the system.

In this paper, we will explore how a human-PR2 team can work together effectively. Two competing goals need to be traded off: maximizing the robot's performance while minimizing human input. A teleoperated robot will perform poorly at complex tasks when the human controls have many degrees of freedom or there are long communication delays. In such scenarios a high level of autonomy would be preferable. The human may merely function as a limited resource for the robot, providing information that may be used to close a planning, control or perception loop. We will explore the concept of *shared autonomy* in the context of perception and grasping. Consider the following scenario: The user issues the command, ”fetch my cup from the kitchen.” The robot would navigate to the kitchen and query the user to select the cup based on a camera image. The robot would then be able to autonomously grasp the cup and carry it back to the user. This human-robot team maximizes the strengths of the robot (data acquisition, navigation, planning) and the strengths of the human (cognition, reasoning).

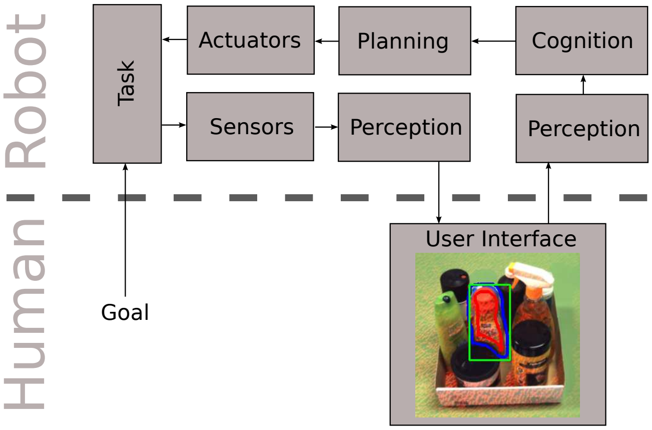


Fig. 1. The shared autonomy system model for robotic manipulation. A human is included to close the perception loop through collaborative object selection.

The PR2's physical capabilities make it well suited for tasks related to mobile manipulation since it has a mobile base, two high degree of freedom arms, and large suite of sensors. In fact, the hardware is not a anymore limiting factor for many complex robotics tasks. Additionally, the PR2 is a very high dimensional system that is difficult to remote control. By combining the PR2's physical capabilities, the ever-growing automated functionalities, and a human operator, we present a system capable of performing mobile manipulation tasks more robustly. We demonstrate a robot that exhibits human like intelligence while only requiring a small portion of a human operator's attention.

II. Related Work

Autonomous object manipulation has been widely explored on recent work. Such approaches perform grasp selection using shape primitives [5], matching of known object models to sensor data [6], or solely by learning grasp points directly from 2D images [7]. In general, to complete object-specific grasping tasks manipulation approaches would need to be combined with an object detection or segmentation algorithm. There is a large literature on the problem of image segmentation [8] and range image segmentation [9], but most previous methods have a limited ability to extract objects

B. Pitzer, M. Styer, C. Bersch, C. DuHadway, and J. Becker are with Robert Bosch LLC at the Research and Technology Center North America, Palo Alto, CA 94304, USA

from a cluttered scene. A large source for failures in object manipulation tasks is the lack of robustness of object detection and grasp selection algorithms. Specifically, systems that rely exclusively on one sensor modality tend are likely to encounter problems more frequently [10]. In this paper we do not add another automatic grasping or segmentation algorithm. Instead, we present a collaborative system (robot-human) in order to overcome current limitations of automatic approaches.

Currently, all robotic systems rely on some combination of human and machine intelligence. How to best coordinate these two sources of intelligence has been studied under a variety of terms including human-robot interaction (HRI), semi-autonomous control, collaborative control, mixed initiative interaction, adjustable autonomy and shared autonomy. In this paper we use the term shared autonomy.

The general approach of using human intelligence to solve ”low-level” computational problems has matured into a robust community of human computation markets. Amazon’s Mechanical Turk is perhaps the best known of several commercial platforms. Surprisingly, the market that comes closest to meeting the requirements of a real-time robotics application is the captcha solving markets which provides solutions to image recognition problems in real-time (median response time of 10s) at large scale (1,000,000 images per day) and low cost ($0.002 per image) [11]. For experimental purposes we used an on-site human operator. However, our approach could be extended to use these human computation markets. Our platform uses $1 to $2 of electricity per day. The same amount spent on human computation would buy 500 to 1,000 real-time solutions to image recognition tasks.

The original and still most common approach to shared autonomy in robotics applications is to assign human operators to supervisory or high level functions and machine intelligence to lower level functions. Sheridan’s widely cited description of ”levels of autonomy” [12] implicitly encodes this approach. This division has been successfully applied in many applications, especially in exploration and navigation tasks [13].

Goodfellow et al. [14] integrated user feedback into a robot to help with action selection. In this case the robot collects environment information, analyzes the scene for objects, asks the human for the appropriate action and then executes that action. In an example task the robot identifies a number of bottles and then a human operator selects which bottle the robot should pick up.

Our system relies on human intelligence to solve a set of low-level perceptual tasks and uses machine intelligence for the remaining high and low level functions. This paper will detail the advantages of this approach. In brief, out shared autonomy concept allows humans to solve problems which are very difficult for current machine intelligences. A similar approach has been previously presented in [15]. Shibuya et al. presented a segmentation method, called CD-matting, which can extract an object in complicated real-world visual situations using a simple scribble input. They showed the effectiveness of a user guided object selection by extending an existing matting technique to deal with color and range data. While their methods performs well for cluttered scenes without occlusions, it remains unclear if a robot can manipulate the extracted objects.

III. Mobile Manipulation

The intrinsic nature of mobile manipulation is that it combines a large number of sub-tasks, many of them active research areas in their own right now. Consider the simple manipulation task of grasping an object and placing it at a different location. An autonomous solution for this task implies the ability to find the object, infer grasp points taking into account the arm and hand kinematics, plan a path to the grasp position, and control the arms. Mobile manipulation additionally implicates the ability to localize, plan a path for the base, and navigate. Robots are now capable of carrying out many of those functions with a varying degree of robustness.

In our definition, shared automation refers to the full or partial replacement of a function that has to be carried out by the robot. We classify this approach as robot-centric [16] because it takes existing robotics algorithms as a basis to solve most of the task and, at limited times, a person functions as a peer to help the robot complete a task. In other words, the human is treated as a limited source of information or processing capability, similar to the automated system components. The partial replacement of single functions implies that the level of automation is not all or none, but varies on scale. For example, Sheridan and Verplank [12] suggested a 10-point scale on which a human operated function receives the lowest score and an entirely automated function receives the highest score.

For any given task, different system functions can and should be automated to differing degrees. For example, robot navigation in indoor environments has been shown tremendous advances in the past decades and is now at a point where robots, when equipped with the right sensors, can robustly operate in large environments over a long period of time. However, the ability to recognize a target object, segment it from the background and compute the desired grasp are at this point very complex tasks with no principal solutions available. Given the mobile manipulation task and the technical capabilities of a system, the question is: which functions should be automated and to what extent? Parasuraman et al. [17] proposed a model for types and levels of automation that provides a framework and an objective basis for making such choices. They propose four function classes: (1) information acquisition, (2) information analysis, (3) decision and action selection, and (4) action implementation. We adopt Parasuraman’s classification and cast the mobile manipulation task into this framework. While Parasuraman’s approach is human centric, aiming at introducing automation to replace functions that were previously carried out (partially or fully) by a human operator, our approach aims to supplement the robot whenever its capabilities are inadequate or ill-suited for the function.

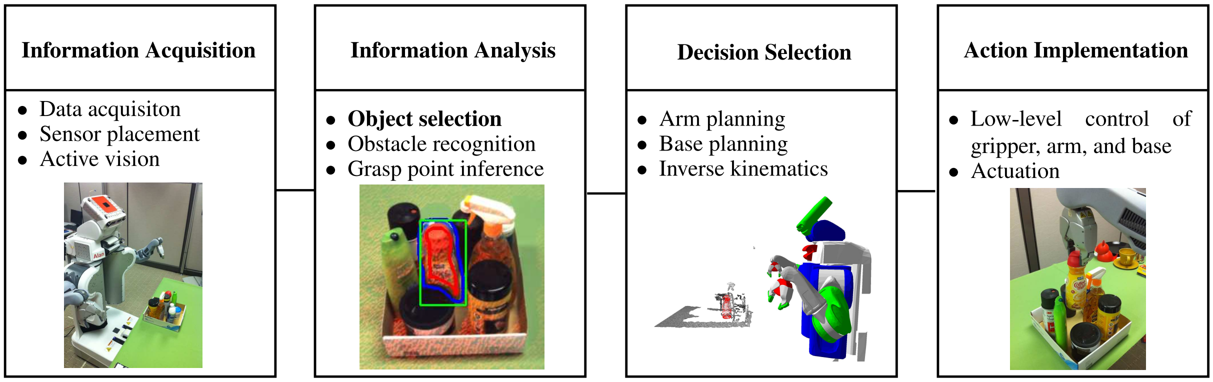


Fig. 2. Sheridan's four-stage model of human information processing applied to the mobile manipulation task of retrieving an object from a table. The *object selection* could be supplemented by a human peer creating a shared autonomy perception system.

The category of **information acquisition** refers to every function that is related to sensing and registration of sensed data. This includes the acquisition of raw sensed data, such as distance measurements, photographs, tactile events, etc., as well as providing information about the state of the robot and its environment, such as a map or object models. For a mobile manipulation task, this category can typically be automated efficiently and robustly.

Information analysis involves cognitive functions such as object recognition and inferential processes. For autonomous agents, one main challenge is to detect instances of objects in sensed data and recognize or categorize them. This is particularly difficult for applications in cluttered environments where sensor data can be incomplete and noisy due to occlusions and outliers. A mobile manipulation robot needs to infer its own state (localization) as well as the state of the environment (mapping) and objects it wants to grasp (object detection and tracking). Automatic localization and mapping (or SLAM) for static environments is considered a solved problem, as far as basic research is concerned [18]. General solutions to reliable multi-class object detection, registration, and tracking remain unsolved, although progress has been made on restricted versions. For example, reliable interpretation of images is still a largely unsolved research problem [19]. Humans, however, are excellent at this task. We can look at a camera image and identify an object in a split second. This makes the object identification and selection task well suited for a shared autonomy application.

Decision and action selection involves selection from among decision alternatives. For the mobile manipulation task this category involves decisions on the navigation part (”where should I go?”) and on the grasping part (”how should I grasp the object?”). Robots, like the PR2, are capable of navigating a complicated, cluttered environment, but a person needs to tell it where to go. With a labeled map, a person can select a location from a set of options.

The fourth category is called **action implementation** and it refers to the actual execution of an action or a decision. This stage involves different levels of machine execution ranging from low level controllers to the physical actuation of the robot's mechanical and electrical components. This category is most commonly accomplished by robots themselves since this is what they were typically built for in the first place: to take over physical activities. Specifically, in industrial settings, the action implementation dominates the capabilities of robotic systems. Interestingly, in most mobile robotic applications the majority of robot operation time is spent in this category.

IV. Perceptual Shared Autonomy

Based on the previous analysis, we choose the *information analysis* as the most promising starting point for a shared autonomy concept. The operator's task will be to supplement the information analysis, namely the object selection, for the robot. All other components will be handled by the robot itself. In this paper, we regard an operator as a remotely located, valuable information source which needs to be managed carefully. In other words, we seek to use the person as little as possible. In Sheridan's terms, letting human operators contribute perceptual analysis can be seen as choosing a low automation level for class (2) while other classes are fully automated.

The considered mobile manipulation task is to pick up an object and place it at a different location. Once the robot is positioned closely to the location of the desired object, a picture is sent to the human peer and he/she is asked to select the object. In our system, we use an interactive object selection method which permits the user to effectively select objects based on color images and point clouds. This is accomplished by drawing a rectangle around it and performing simple strokes to mark object and background areas. Once the user has finished the selection task, control is handed back to the robot. Based on the selected image region, the interactive object selection method automatically finds corresponding 3D points and matches those points to an object model. The model includes pre-computed grasps [5] which are used to plan the grasp arm motion. The grasp is then automatically executed by the robot.

V. Interactive Object Selection

The interactive segmentation algorithms using graph-cuts [20], [21], [22] have been proven to be powerful tools for accurate segmentation of objects from background. Those methods always segment the image into two regions: foreground and background. For robotic applications those methods have been widely disregarded since they cannot be automated. The user must supply the critical parameters to the algorithm like a rectangle around the object in the scene [22] or seed points from the desired foreground and the background [20], [21]. Typically, graph-cut algorithms work on the basis of pixel intensity or color distributions. Therefore, a major problem is: if some part of the foreground object has a color distribution similar to the image background, that part will also be assiged to the background. Through more user input, such as more strockes distinguishing fore- and background, the algorthim can typically recover from those cases. In this section, we briefly discuss the original graph-cut algorithm and present our generalization to the segmentation of sensor data from color and range images.

A. *Graph-cuts for image segmentation*

Boykov et al. [20] described the image segmentation problem as a directional flow graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. The node set $\mathcal{V}$ is defined by the image pixels as well as two nodes called terminal nodes which represent the two classes ”object” and ”background.” The directed weighted edges $\mathcal{E}$ connect pixel nodes in a local neighborhood and all pixel nodes to both terminal nodes. A segmentation is performed by finding the minimum cut of this graph, which is a subset of edges that divide the graph into two parts: the object- and the background-part, hence resulting in a binary segmentation. The cost of a cut being the sum of the cost of its edges. This method is semi-automatic as the user needs to select two sets of image pixels $\mathcal{V}_o$ and $\mathcal{V}_b$, containing some pixels of the object and the background respectively.

Consider an image $\mathcal{I} = (z_1, \ldots, z_N)$ as a vector of pixels, where z_i are the intensity gray values. Let the set $\mathcal{N}_8$ be all pixel pairs $\{z_i, z_j\}$ of all 8-neighborhoods. The segmentation is expressed as a vector of label values $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_N)$ at each pixel, where $\alpha_i \in \{"object", "background"\}$.

The graph-cut image segmentation algorithm [20] defines a cost function E such that its minimum corresponds to a ”good” segmentation. A good segmentation is guided both by the observed foreground and background color distributions and a smooth segmentation. This is captured by a Gibbs energy of the form:

$$E\left(\boldsymbol{\alpha}\right) = \sum_{z_i \in \mathcal{I}} R\left(z_i\right) + \gamma \sum_{\{z_i, z_j\} \in \mathcal{N}_8} B\left(z_i, z_j\right) \ . \qquad (1)$$

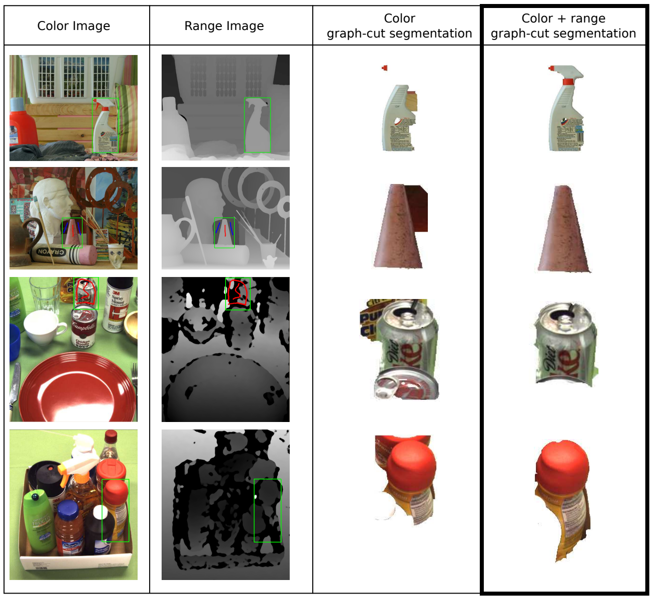


Fig. 3. Interactive object selection using graph-cut segmentation. An initial segmentation is performed by drawing a rectangle around the desired object (green) and in difficult cases, additional strokes for foreground (red) and background (blue). The initial segmentation is then automatically refined using graph-cuts on the color and range constraints. The results of this segmentation are shown in the fourth column. In comparison segmentation results using color constraints alone are presented in the third column. The first two images are courtesy of the Middlebury Stereo Dataset [23].

The term $R(.)$, commonly referred to as the regional term, expresses how the pixel z_i fits into given models of the object and background. All $\mathcal{V}_o$ seeds are connected to the object node and all background seeds $\mathcal{V}_b$ are connected to the backgroud node. Those links form the regional term and the cost is based on how the intensity z_i fit into given intensity models (e.g., histograms) of the object and background. The term $B(.)$, known as the boundary term, reflects the similarity of the voxels z_i and z_j. Typically, the edge weight between neighboring pixels is choosen to be $B\left(z_i, z_j\right) \propto \exp(-\beta |z_i - z_j|)$. This weight function forces the segmentation boundaries at places with high intensity gradient.

The graph is now fully defined and a segmentation can be estimated as the global minimum: $\hat{\boldsymbol{\alpha}} = \text{argmin } E\left(\boldsymbol{\alpha}\right)$. Thus, the minimum cut, is the cut with the minimum cost and can be computed in polynomial time using the max-fow algorithm [20].

B. *Graph-cuts for color and range image segmentation*

In contrast to classical image processing applications, we have different types of information available for robotic tasks, such as color images from a camera and range images from a laser range finder or stereo-vision. In our segmentation method, we seek to combine data from multiple sources and gather that information in order to achieve a better segmentation. This will be more efficient and potentially more accurate than if they were achieved by means of a single source. Specifically, combining both color and range constraints provides tighter constraints on the system than either color or range constraints used separately.

Assuming the color data is registered with the range data, the pixels $\boldsymbol{z}_i = \{\boldsymbol{c}_i, d_i\}$ are now taken to consist of a vector of RGB values and a depth value. The regional term $R(.)$ and the boundary term $B(.)$ are defined separately for color and depth. Similar to [22], we use Gaussian mixture models (GMMs) θ to express the color distributions for the background and foreground as a mixture of K Gaussian distributions. In order to deal with the GMM tractably an additional vector $\boldsymbol{k} = (k_1, \ldots, k_N)$ is introduced, with

$k_n \in \{1, \ldots K\}$, assigning, to each pixel, a unique GMM component. The resulting regional term for the color components is:

$$R_{\boldsymbol{c}}\left(\boldsymbol{c}_i\right) = -\log p\left(\boldsymbol{c}_i | \alpha_i, k_i, \theta_i\right) - \log \pi\left(\alpha_i, k_i\right) \ . \tag{2}$$

Here, $p(.)$ is a Gaussian probability distribution, and $\pi(.)$ are mixture weighting coefficients. The means and covariances of the Gaussian components for the background and foreground distributions are formed from the respective GMM components. The regional term for the range information is simply the fit of d_i to a histogram:

$$R_d\left(d_i\right) = -\log \ \text{hist}\left(d_i, \alpha_i\right) \ . \tag{3}$$

The resulting combined regional term for a pixel $\boldsymbol{z}_i$ is formulated in a similar fashion to a L2 norm:

$$R\left(\boldsymbol{z}_i\right) = \sqrt{R_{\boldsymbol{c}}\left(\boldsymbol{c}_i\right)^2 + R_d\left(d_i\right)^2} \ . \tag{4}$$

In a similar manner, we formulate the joint boundary term $B(.)$. For the color components the boundary term [22] is given as follows

$$B_{\boldsymbol{c}}\left(\boldsymbol{c}_i, \boldsymbol{c}_j\right) = \delta\left(\alpha_i, \alpha_j\right) \exp\left(-\beta_{\boldsymbol{c}} \left\|\boldsymbol{c}_i - \boldsymbol{c}_j\right\|^2\right) \tag{5}$$

with

$$\delta\left(\boldsymbol{c}_i, \boldsymbol{c}_j\right) = \begin{cases} 1 & \text{if } \alpha_i \neq \alpha_i \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Here $\beta_{\boldsymbol{c}}$ is a constant chosen to be proportional to the contrast of a selected image region. For the range components, the boundary term is basically unchanged from the color case:

$$B_d\left(d_i, d_j\right) = \delta\left(\alpha_i, \alpha_j\right) \exp\left(-\beta_d \left|d_i - d_j\right|\right) \tag{7}$$

Again, both terms are combined to form the joint boundary term:

$$B\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right) = \sqrt{B_{\boldsymbol{c}}\left(\boldsymbol{c}_i\right)^2 + B_d\left(d_i\right)^2} \ . \tag{8}$$

The minimum cut optimization for the new graph can be performed exactly the same way as in the original algorithm [20]. See Fig. 3 for an example of interactive graph-cut segmentation using color images compared to our approach using color or range constraints.

VI. Experiments

A. Hardware Setup

The robot platform used for the experiments in this paper is the PR2 personal robot [24], a two-armed robot with an omnidirectional base. Equipped with two 7 degrees of freedom compliant arms, the PR2 is designed for compliant interaction with the environment. The PR2's head is a pan-tilt platform equipped with a high resolution 5 megapixel camera and two stereo camera pairs with different parameters for narrow and wide field of view receptively. A strong texture projector supplements the stereo system to reduce dropouts on objects with little or no texture. The narrow field stereo sensor is used in this work to generate a 3D point cloud of the scene since it provides dense and accurate depth information for objects close to the robot. The 3D point cloud is then projected into the high resolution camera to form a virtual range image which is aligned with the color image through calibration. The software controlling the PR2, called ROS [25], encompasses many of the components needed for mobile manipulation: hardware drivers, controllers, perception algorithms, motion planning, high-level planning, etc.

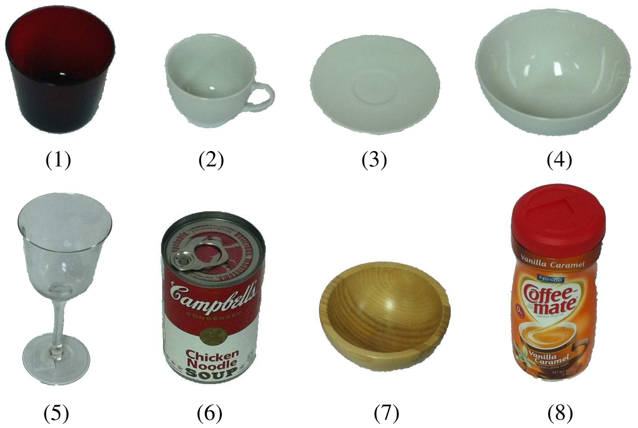
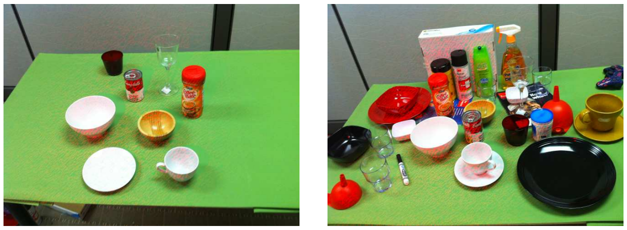


Fig. 4. The objects used for the mobile manipulation experiments are typical household items: red glass (1), coffee cup (2), saucer (3), white bowl (4), wine glass (5), soup can (6), wood bowl (7), coffee creamer (8).



(a) Scene 1. (b) Scene 2.

Fig. 5. Two scenes used for the comparison of object selection approaches.

B. Comparison of object selection approaches

In the first experiment, we are concerned with comparing the performance of our shared autonomy object selection approach with two state of the art automatic object selection algorithms. The task was to select a desired object, pick up the object and place it at a different location in front of the robot. The objects used in our experiments are depicted in Fig. 4. The input data for algorithms consists of the desired object class, a range image, and a color image. The different object selection approaches were tested on the PR2 personal robot [24].

The first algorithm is the ROS tabletop object detection[1]. The tabletop detection first estimates the table by finding the dominant plane in the point cloud using RANSAC, then clusters all points above the table to identify individual objects, and finally applies a simple iterative fitting technique to match the clusters to a model of the desired object in a database. If a good fit is found, pre-calculated grasp points

[1] http://www.ros.org/wiki/tabletop_object_detector

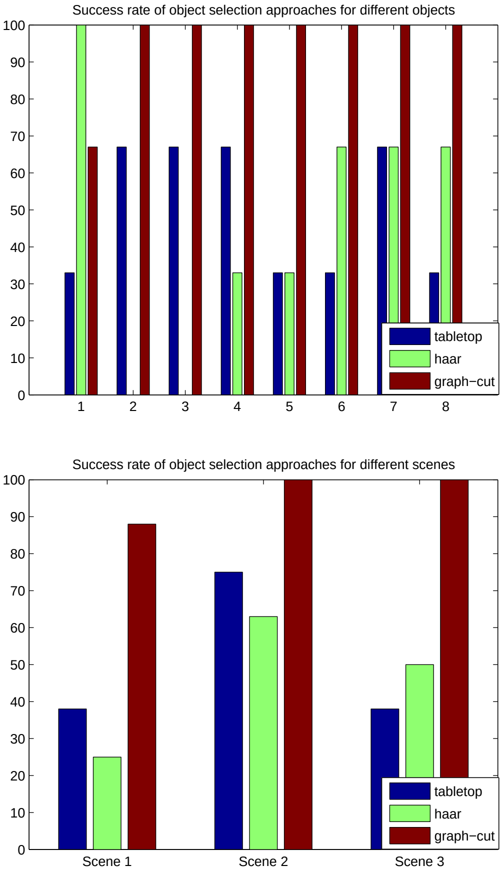


Fig. 6. Success rates for different object selection approaches. The top figure shows the success rates over all tested objects while the bottom figure presents success rates over all tested scenes.

are used to grasp the object. Note that this approach detects objects purely based on point clouds. The second automatic algorithm is the well known Haar feature-based cascade classifier by Viola and Jones [26]. This vision based classifier was trained with a few positive sample views of each object as well as a number of negative samples. After training, a search window is moved across incoming images and the classifier is applied at every location at different scales. If a region is likely to show the object the classifier returns the corresponding bounding rectangle. For automatic object selection, the stereo point cloud is projected into the camera image and the same iterative fitting technique used for the tabletop detection is applied to all points that fall into the bounding rectangle returned by the classifier. Recall that our interactive selection approach also uses the projected stereo points for model based fitting. For the evaluation of the three object selection approaches, we used a number of different scenes which contained all test objects and a variable number of other objects. Two of those scenes are depicted in Fig. 5.

Fig. 6 shows the success rates over all tested objects. It shows the fraction of times the PR2 arm was able to physically pick up the object. On average, our shared autonomy approach was able to pick up the objects 96% of the time whereas the ROS tabletop object detection and the Haar detection only average on 44% and 46% respectively. For a human, the object identification and selection task based on the robot's camera images is trivial, and this part was successful at all times. The only failures in the human assisted approach we observed were related to the model fitting procedure which failed to find the correct fit. The failure cases in the point cloud based tabletop object detection were mostly in the clustering stage which merged points of two or more objects into one cluster. This problem is amplified in scenes where objects are arranged in a close proximity (Scene 2). The Haar detection failures were mostly related to a wrong classification. Specifically, partially occluded objects posed great challenges for this vision based approach. This is not surprising, since partial occlusions were not included in the training set for the Haar detector. A different view on the data is provided in second graph of Fig. 6 which shows the success rates over different test scenes.

In addition to reporting success rates, we also report the time a person or the computer spent on the object selection task. On average, the tabletop detection took 5.43 seconds and the Haar detection only 0.13 seconds. An untrained person completed the interactive object selection in 17.57 seconds which is more than 100 times longer than the Haar detection.

An interesting observation is that certain test objects worked better with certain object selection methods. For example, the Haar detector confused the white coffee cup (2) with the white bowl (4) on many occasions since they look alike very much in camera images. In contrast, the same classification mistake occurred almost never with the tabletop object detection approach because in 3D those objects can be clearly distinguished just by their size and shape. In return, the tabletop object detection had significant problems with translucent objects, such as the wine glass (5), where only very few 3D points are reconstructed by the stereo vision system. The shared autonomy approach can handle both cases since: 1) a person can easily separate the object classes even in very difficult situation such as large occlusion or dim light conditions and 2) selecting translucent objects works in the same manner as opaque objects and only very few points are necessary for the automatic model fitting procedure to ground the object in space.

C. *Challenging object selection situations*

In the second experiment, we use our shared autonomy object selection approach to accomplish tasks which would be very challenging for current automatic methods. The tasks included grasping an object which is inside another object, picking a straw out of a soda can, and retrieving an object from a box with many other objects. See Fig. 7 for the described scenarios. Those tasks are challenging for object selection algorithms in many ways: stacked or combined objects are typically not modeled in automatic algorithms, most perception systems are ill-suited to reconstruct straws, and cluttered environments with largely occluded objects

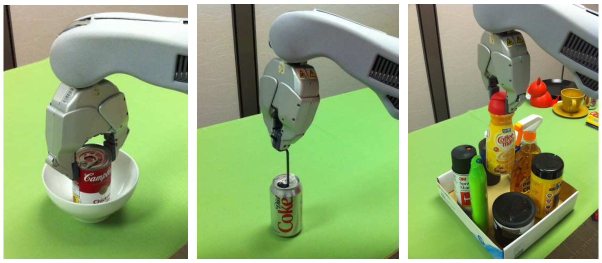


Fig. 7. Examples for interactive object selection in challenging situations.

are extremely challenging for automatic object detection algorithms.

In many instances, the interactive object selection was able to select and pick up the mentioned objects. Perceiving very small and thin objects (straws) is a difficult problem for our stereo vision system since the stereo algorithm was tuned to reconstruct rather smooth surfaces instead of thin and small objects. The interactive object selection worked surprisingly well in cluttered environments, even for heavily occluded objects. Even small parts of occluded objects can be picked up easily by a human operator and selected with our interactive object selection method.

VII. Summary and Conclusion

Insufficient robot perception is a major road block for many real-world robotic applications. In this paper, we presented method for robotic perception which queries a person to solve the difficult task of object selection for the robot. We showed that a human-robot team can work together effectively solving a typical object manipulation task. The presented method allows a person to select an object by drawing rectangles and performing simple strokes to separate objects from background areas in color images. In various experiments with the PR2 robot we showed that this shared autonomy system performs more robustly than state-of-the-art automatic object selection algorithms, and that it is capable of tasks which are very challenging to accomplish by an autonomous agent. For an untrained user, the selection time for one object is approximately 20 sec. The experiments show a clear correlation between the success rate of certain object classes and the tested automatic detection methods. A future direction could be to apply Value-Of-Information theory to the object selection task, such as suggested in [27]. This will allow the robot to decide when to query operators, i.e., humans are only queried if the expected benefit of their interaction exceeds the cost of obtaining it compared to using one or many automatic selection approaches.

References

[1] T. W. Fong, C. Thorpe, and C. Baur, “Robot, asker of questions,” *Robotics and Autonomous Systems*, 2003.

[2] T. Kaupp and A. Makarenko, “Measuring human-robot team effectiveness to determine an appropriate autonomy level,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 2146–2151.

[3] G. Podnar, J. Dolan, A. Elfes, and M. Bergerman, “Multi-level autonomy robot telesupervision,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[4] Anybots, “QB.” [Online]. Available: http://anybots.com

[5] A. T. Miller and P. K. Allen, “Graspit! a versatile simulator for robotic grasping,” *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[6] S. Srinivasa, D. Ferguson, J. M. Vandeweghe, R. Diankov, D. Berenson, C. Helfrich, and K. Strasdat, “The robotic busboy: Steps towards developing a mobile robotic home assistant,” in *International Conference on Intelligent Autonomous Systems (IAS)*, 2008.

[7] A. Saxena, J. Driemeyer, J. Kearns, and A. Ng, “Robotic grasping of novel objects,” *The International Journal of Robotics Research*, vol. 27, pp. 157–173, 2008.

[8] H.-D. Cheng, X. Jiang, Y. Sun, and J. Wang, “Color image segmentation: advances and prospects,” *Pattern Recognition*, vol. 34, pp. 2259–2281, 2001.

[9] P. Boulanger, “Simultaneous segmentation of range and color images based on bayesian decision theory,” *Computer and Robot Vision, Canadian Conference*, pp. 58–63, 2004.

[10] K. Hsiao, S. Chitta, M. Ciocarlie, and E. Jones, “Contact-reactive grasping of objects with partial shape information,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[11] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. Voelker, and S. Savage, “Re: Captchas understanding captcha-solving services in an economic context,” in *USENIX Security Symposium*, 2010.

[12] T. Sheridan and W. Verplank, “Human and computer control of undersea teleoperators,” MIT Man-Machine Systems Laboratory, Tech. Rep., 1978.

[13] M. A. Goodrich and A. C. Schultz, “Human-robot interaction: a survey,” *Found. Trends Hum.-Comput. Interact.*, vol. 1, no. 3, pp. 203–275, 2007.

[14] I. Goodfellow, N. Koenig, M. Muja, C. Pantofaru, A. Sorokin, and L. Takayama, “Help me help you: Interfaces for personal robots,” in *Proceedings of Human Robot Interaction (HRI)*, 2010.

[15] N. Shibuya, Y. Shimohata, T. Harada, and Y. Kuniyoshi, “Smart extraction of desired object from color-distance image with user's tiny scribble,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[16] T. W. Fong, C. Thorpe, and C. Baur, “Collaborative control: A robot-centric model for vehicle teleoperation,” in *AAAI 1999 Spring Symposium: Agents with Adjustable Autonomy*, 1999.

[17] R. Parasuraman, T. Sheridan, and C. Wickens, “A model for types and levels of human interaction with automation,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 30, no. 3, pp. 286–297, 2000.

[18] U. Frese, “Interview: Is slam solved?” *KI - Künstliche Intelligenz*, vol. 24, pp. 255–257, 2010.

[19] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

[20] Y. Boykov and M.-P. Jolly, “Interactive graph cuts for optimal boundary – region segmentation of objects in n-d images,” in *IEEE International Conference on Computer Vision (ICCV)*, 2001.

[21] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, may. 2002.

[22] C. Rother, V. Kolmogorov, and A. Blake, “”grabcut”: interactive foreground extraction using iterated graph cuts,” *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.

[23] D. Scharstein and C. Pal, “Learning conditional random fields for stereo,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 0, pp. 1–8, 2007.

[24] Willow Garage, “Personal Robot 2 (PR2).” [Online]. Available: http://www.willowgarage.com

[25] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *Workshop on Open Source Software*, 2009.

[26] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Conference on Computer Vision and Pattern Recognition*, vol. 1. Los Alamitos, CA, USA: IEEE Comput. Soc, April 2001, pp. I–511–I–518.

[27] T. Kaupp, A. Makarenko, and H. Durrant-Whyte, “Human-robot communication for collaborative decision making - a probabilistic approach,” *Robots and Autonomous Systems*, vol. 58, no. 5, pp. 444–456, 2010.