A Scooter-Mounted Robot Arm to Assist with Activities of Daily Life

Dian Wang, Colin Kohler, Andreas ten Pas, Maozhi Liu, Holly Yanco, Robert Platt

Abstract—Many people with motor disabilities struggle with activities of daily life (ADLs), limiting their ability to live independently. This paper details a robotic mobility scooter developed to assist with manipulation-based ADLs to increase independence. We present a system comprised of a Universal Robotics UR5 robotic arm, a mobility scooter, five depth sensors, and a user interface which utilizes laser pointers. The system provides pick-and-drop and pick-and-place functionality in open world environments without modeling the objects or environment. We evaluate our system over several experimental scenarios and show an improvement relative to a baseline established for a similar system.

I. INTRODUCTION

The ability to grasp and manipulate objects is one of the most important physical capabilities for activities of daily life (ADLs). However, for the millions of people in the United States currently living with motor disabilities and for elderly people, simple manipulation tasks can be challenging [1]. While there has been prior work (e.g., [2], [3], [4], [5]) in developing robotic manipulation systems to assist people with ADLs, the performance of these systems is below acceptable limits or they require known models of objects that are to be manipulated. For example, our prior work used a Baxter robotic arm in conjunction with a mobility scooter [4]. However, both the average time required to execute a grasp (128s) and the grasp success rate (72%) made it difficult to deploy that system in real assistance scenarios.

The key contribution of this paper is the integration and demonstration of a new robotic mobility scooter based on the grasp detection capability introduced in [6], [7]. Compared to our prior work [4], there are several new innovations. First, we present the idea that the user can manually select grasps from a collection of automatically generated grasps. This enables the user to supervise grasp selection and allows the system to work in complex environments. Second, we introduce pick-and-place functionality, which allows the user to place a grasped object on top of an arbitrary surface in the same orientation as it was grasped. Third, we improve the hardware configuration by replacing the Baxter arm [4] with a UR5 arm and by integrating five Structure IO (StructIO) sensors that enable our system to manipulate objects nearly anywhere in the workspace of the arm. In particular, our new system can grasp objects from the floor – something that our prior system [4] could not do. Finally, our new dual-laser user interface provides more stable laser pointer detection.

We benchmark against prior work [4] using the same experimental protocol for grasping in isolation and grasping in-situ. In grasping in isolation experiments, we demonstrate an average grasp success rate of 93.8%, a 4.2% improvement. When grasping in-situ, we demonstrate an average grasp success rate of 96% in an average time of 69s, a 24% improvement on grasp success rate and a 46% improvement on time when compared to the benchmark. Finally, we evaluate our approach in a new, more complicated, environment with a series of pick-and-place tasks.

II. RELATED WORK

Much of the prior work in this domain has focused on the design and evaluation of human-robot interaction (HRI) methods. While our system also has an HRI method, the focus of this paper is the automated grasping of unknown objects; a study of the HRI is left to future work. In this section, we present grasp success rates for related work, if it was included in the publication.

In 2001, Martens et al. introduced a semi-autonomous

*This work has been supported in part by the National Science Foundation through IIS-1426968, IIS-1427081, IIS-1724191, and IIS-1724257, IIS-1763469, NASA through NNX16AC48A and NNX13AQ85G, ONR through N000141410047, Amazon through an ARA to Platt, and Google through a FRA to Platt.

1College of Computer and Information Science, Northeastern University, Boston, Massachusetts

2Department of Computer Science, University of Massachusetts Lowell, Lowell, Massachusetts
robotic system, composed of a robot arm mounted on a wheelchair with a speech interface [2]. Kemp’s lab developed a system with a laser pointer interface to allow a user to select an object in the world that their mobile robot would retrieve [8], [3], with 90% (10 grasps) and 100% (6 grasps) grasping success rates, respectively. In their experiments, only a limited number of grasp attempts were performed and the arm could only approach grasps from top down. In later work, they provided additional HRI solutions including an ear-mounted laser pointer and a touch screen interface [9]. Achic et al. proposed the combination of a hybrid brain-computer interface and an electric wheelchair equipped with a robotic arm to facilitate navigation and manipulation tasks [10]. Grice et al. demonstrated an assistive robotic system, in which a PR2 robot can be teleoperated for manipulation tasks via a modern web browser and a single-button mouse [11]; this system requires direct end-effector control for grasping from the user, while our system provides automatic grasp detection.

Grasp detection is a key piece of our system. We use GPD [6], [7], a grasp detection system with high grasp success rates that is easily adapted to unstructured grasp scenarios. However, it is important to point out several related pieces of work in the grasp detection literature. Perhaps the most well known is the work of [12] who developed a closed-loop grasp system using large amounts of real robot experience. Another related work is [13] who developed a grasp detection system that takes depth images as input. Although both of these methods achieve relatively high grasp success rates (approximately greater than 90%), they are tuned for the bin-picking scenario and as such would be difficult to apply in the current application. Other related grasp detection research proposes a method of adapting grasps from canonical models to novel objects [14] or focuses on learning in simulated domains [15], [16].

Our prior work [4] presented a system comprised of a robotic arm from a Rethink Robotics Baxter robot mounted on a mobility scooter. A laser pointer device was used to locate the target object and GPD was used to detect grasps on that target. Although the system was able to perform grasping tasks in an open environment, there were two primary issues: the time required for each grasping task was too high and the grasp success rate was too low. The average time needed to execute a grasp was 128s and the maximum time was 374s. The primary cause of this long execution time was the use of a single wrist-mounted sensor which required movement of the arm while generating the point cloud data used for laser and grasp detection. Although the system had a high grasp success rate (89.6%) in isolation, it had a significantly lower success rate (72.1%) in the more realistic mobile scenarios.

III. HARDWARE

The system is comprised of a UR5 robotic arm (6-DOF), equipped with a Robotiq 2-finger 85 gripper, mounted on a Merits Pioneer 10 mobility scooter as shown in Fig. [1]. The perception subsystem is made up of five StructIO depth sensors mounted between the robot and the scooter (Fig. [1]). Specifically, there are four sensors mounted near the handlebars (two on the left and two on the right) and an additional sensor mounted higher up which can see elevated surfaces like the top of a shelf or a tall table. Using this five-sensor setup, the vision system can detect objects anywhere in a one-meter tall area in front of the scooter.

A dual laser pointer device, a ten-inch monitor, and an X-keys XK-4 stick make up the physical user interface (Fig. [2]). We developed a graphical user interface (GUI) using RViz [17], which displays the output of our system including the perception results, the status of the system, and current actions available to the user. The user interacts with this interface by using the key stick. The dual laser pointer allows the user to select the next target (either the object to pick or the location to place) for the system. The device consists of two (5 mW 650 nm) red lasers. As the system detects this pair of laser points in the environment, the user can use a single pointer to position the device on the target and then use both pointers to confirm.

A workstation is mounted on the back of the scooter and is connected to the robot, sensors, and user interface. The workstation consisted of a 3.6 GHz Intel Core i7-6850K CPU (six physical cores), 32 GB of system memory, and a Nvidia GeForce GTX 1080 graphics card. The perception subsystem, grasp detection subsystem, planning subsystem, and visualization subsystem are all implemented as ROS [17] nodes on the workstation.

IV. PERCEPTION

A. Point Cloud Generation

Both the grasp detection subsystem and the planning subsystem use point cloud information as the primary input, so it is important to obtain the most complete point cloud possible of the workspace. Our system attempts to achieve this using five StructIO sensors: two near each handlebar and one mounted on a post, approximately five feet above the ground. The four sensors at the handlebars cover most of the workspace in front of the scooter above the floor with at least two views. The fifth sensor on the post covers the floor with a top-down view. Coverage of the workspace with multiple views is important for both obstacle avoidance and
for grasp detection; as shown by [7] and [6], grasp detection
accuracy can improve by as much as 9% simply by adding
one additional view. Due to interference between sensors
when illuminating simultaneously, the point cloud from each
sensor has to be acquired while the others are off.

B. Laser Pointer Detection

A dual laser pointer device is used to identify the target for
the next pick or place action. The system detects the position
of the laser pointers by reading the infrared radiation (IR)
image from StructIO sensors. Since our device consists of
two parallel laser pointers, each potential location is detected
by finding two nearby high-value areas within each IR image.

For each time frame, the system detects at most one
potential laser pointer location from each of the five sensors.
These areas of interest are found by extracting all pixels
with an IR value above 10 (unless there are fewer than
10 of these pixels, in which case the 10 pixels with the
highest IR values are extracted). These pixels are clustered
with hierarchical clustering [18] such that the pixels in each
cluster have no greater cophenetic distance than three. After
removing clusters which are too small (< 2 pixels) or too
large (> 9 pixels), the distances between each pair of clusters
is measured. If this distance is between 10 and 50, then
the center of the two clusters is marked as a potential laser
pointer location. After detecting all the potential locations,
one location is randomly selected as the potential laser
pointer location for the current sensor.

Once potential locations have been detected for each
sensor, the system will decide that a successful detection
has occurred if one of the following is satisfied: (1) Two or
more sensors detect potential laser pointer locations such that
the max value of the variance of their x, y, z components
is less than or equal to 0.01 meters. (2) Only one sensor
detects a potential laser pointer location that is at most 0.01
meters away from a saved potential laser pointer location in
the previous time step. Saving the previous potential laser
pointer location helps the system increase the detection rate
when only one of the sensors can view the laser pointers.

V. GRASP DETECTION

A. Preprocessing

The goal of point cloud preprocessing is to eliminate from
consideration those parts of the environment that can be
safely excluded using standard methods. First, we remove
all planes that are roughly horizontal with respect to the
vehicle from the cloud. We use RANSAC [19] parameterized
with a distance threshold of 0.015 meters and a maximum
angle relative to the horizontal of five degrees. Planes are not
removed unless they contain at least 13k inliers. We eliminate
at most 70% of the points in the cloud this way.

After removing horizontal planes, we attempt to segment
individual objects using Euclidean cluster extraction [20]
with a distance threshold of 0.005 meters. The goal here is
to eliminate all objects from consideration that can easily be
segmented. We eliminate all clusters except the one closest
to the point of interest found using our laser point detection
subsystem. If the remaining cluster has fewer than 500
points, we skip this step and simply return all points within
a 0.1-meter radius ball centered at the laser pointer position.

B. Grasp Pose Detection

After point cloud preprocessing is complete, we detect
feasible grasps using a publicly available grasp detection
system known as GPD [6], [7], developed by our research
group. GPD detects 6-DOF robotic hand poses that are pre-
dicted to be feasible grasps. The system takes two different
point clouds as input: the segmented cloud as described
above and the full cloud. GPD uses the segmented cloud
to seed candidates for grasp detection and the full cloud
to ensure that the detected grasps do not collide with the
environment. Because the seeded grasps are drawn only from
the segmented cloud, the system only detects grasps in the
vicinity of the segmented object.

VI. GRASP AND PLACE SELECTION

After high-scoring grasp candidates are found, the system
filters out all grasps that do not have a collision-free inverse
kinematic (IK) solution [21]. To select a grasp to execute
from the set of remaining feasible grasps, the user can
either defer to a built-in set of heuristics or select the grasp
manually.

A. Automatic grasp selection

In automatic grasp selection, we prioritize the grasps using
an objective function equal to the product of the following
five factors:

\[ C = C_w C_j C_a C_s C_p \] (1)

\[ C_w = 1 - \frac{\max(0, W_d - \min(|G_w - W_{\min}|, |G_w - W_{\max}|))}{W_d} \] (2)

\[ C_j = \frac{(J_i - G_j) \cdot (J_i - G_j)}{J_i \cdot J_i} \] (3)

\[ C_a = \begin{cases} 
0.5 + 0.5|G_a[1]|, & |G_2[2]| > 0.8 \\
1, & \text{otherwise}
\end{cases} \] (4)
Where $G_a[1]$ is the $y$ component in the approach vector (the $y$ axis in world frame is pointing forward) and $G_x[2]$ is the $z$ component in the axis vector.

$C_s$ encodes a preference for grasps that are nearby the segmented cloud as this decreases the possibility that the object will slide away during grasping. Let $G_b$ be a fixed point relative to the grasp position, $l$ be the gripper length, and $S$ be the sample cloud. The sample distance cost $C_s$ is computed as:

$$C_s = 1 - \frac{\min(l, \min_{p \in S} \|p - G_b\|)}{l}$$

(5)

Finally, $C_p$ penalizes grasps far away from the detected laser pointer position. Let $p$ be the laser pointer position and $th$ be a distance threshold (0.05m in our experiments). The position distance cost $C_p$ is computed as:

$$C_p = \exp(-10 \times \max(0, \|p - G_b\| - th))$$

(6)

All reachable grasps are sorted by the result of the objective function, from largest to smallest.

B. Manual Grasp Selection

Although automatic grasp selection often works well, there are a few scenarios where it has problems. First, in dense object setups, it is possible that the segmented point cloud contains multiple objects and a grasp on the wrong object is selected. Second, the system may attempt to grasp the table, shelf, or some other part of the environment. Third, if the user is performing pick-and-place, they may have a preference for grasping the object in a particular way. The option for manual grasp selection enables the user to overcome these problems by specifying to the system exactly how to grasp the object. Note that the grasp detection system is still active here – the only difference is that after grasp detection is occurred, the user may select which detected grasp is most suitable instead of relying on the heuristics described earlier.

During manual grasp selection, the set of feasible grasps are visualized on a small monitor mounted near the scooter handlebars (see Fig. 1) and the user can toggle through them using the key stick to select the desired one. In order to facilitate this process, we cluster the detected grasps to remove near-duplicates using hierarchical clustering [18] on the 3-DOF position of the grasps and then displaying these clusters. The max Euclidean distance inside each cluster of grasps is 0.02 meters. After the user makes a selection, all detected grasps are sorted based on the Euclidean distance from the selected grasp.

C. Place Selection

When in pick-and-place mode, grasps are generated and executed in the same manner as above. However, after the grasp is selected we measure the distance between the grasp and the surface the object is on. This offset will be added to the next laser pointer detection in order to adjust to the height the object was grasped at. Once the grasp has been executed, the user is free to reposition the scooter in front of the surface they wish to place the object onto. The placement
TABLE I
RESULTS FOR GRASPING IN ISOLATION AND GRASPING IN-SITU

|                      | Success Rate | Average Time |
|----------------------|--------------|--------------|
|                      | Detection    | Planning     | Grasping  | Cloud | Laser | GPD | Filter | Planning | Execution | System | Driving | Total | Total Incl Failures |
| Work of [7] (Dense Clutter) | -            | -            | 93%       | -     | -     | -   | -      | -        | -         | -      | -       | -     | -               |
| Work of [4] (Isolation Exp) | 87.8%       | 100%         | 89.5%     | -     | -     | -   | -      | -        | -         | -      | -       | -     | -               |
| This Work (Isolation Exp)   | 92.3% (96/104) | 100% (96/96) | 93.8% (90/96) | 8.0s  | 5.8s  | 7.4s | 1.6s   | 2.4s     | 24.2s    | 24.2s  | 51.7s  | -     | -               |
| Work of [4] (In-Situ Exp)   | 89.4%       | 100%         | 72.1%     | -     | -     | -   | -      | -        | -         | -      | -       | -     | 128s            |
| This Work (In-Situ Exp)     | 85.3% (58/68) | 86.2% (50/58) | 96% (48/50) | 8.0s  | 6.0s  | 4.7s | 1.6s   | 2.9s     | 24.3s    | 50.7s  | 11.4s  | 69s  |               |

Detailed time spent does not include system loss time, e.g., time for user pressing the key.

2) Results: Table I shows the results from this experiment (the rows labeled Isolation Exp). We attempted to remove all objects from the table 15 times. Each time, the table was initialized with six objects placed in random locations as described above (a total of 90 objects to be grasped). It took our system 104 attempts to grasp these 90 objects where 96 out of 104 laser detection attempts succeeded and 90 out of 96 (93.8%) grasping attempts succeeded. On average, each grasp took 51.7 seconds to perform. This includes 8s to acquire the point cloud, almost 6s to find the laser pointer, and 24s to move the arm to execute the grasp. Compared to [4], this constitutes a 4.5% improvement in laser detection success rate and a 4.2% improvement in grasp success rate.

B. Evaluating Grasping In-Situ

1) Experimental Protocol: To test our system in a real-world scenario, we used the open kitchen area shown in Fig. 4a. We performed five trials. In each, ten objects were randomly selected from the object set and placed in randomly assigned positions. Three were placed on a 73 cm tall table, three were placed on a 31 cm tall table, two were placed on the top shelf of a bookshelf (100 cm high), and two were placed on the middle shelf of the bookshelf (69 cm high). Certain objects were not allowed to be placed on the middle shelf as they could only be grasped from the top down and the arm could not fit inside the shelf. These trials were run with automatic grasp selection in pick-and-drop mode by an expert user. The sequence in which the objects had to be grasped was randomly generated for each trial. At the start of each task, the scooter was driven to Start Point 1 (seen in Fig. 4a) and then up to the next object in the sequence to be grasped. We evaluated the system according to the same metrics used in Section VII-A along with the time for each task, which includes driving the scooter up to the target object and the time to successfully grasp the object.

2) Results: Out of the 50 tasks (5 trials each with 10 objects), 48 succeeded, giving us a task success rate of 96%. The two failures were due to objects falling over on the middle shelf during the experiment, making it impossible for the system to grasp them. As is shown in Table II (rows labeled In-Situ Exp), 58 out of 68 laser detections succeeded, 50 out of 58 motion plans succeeded, and 48 out of 50 grasping attempts succeeded. This is a 24% improvement on the grasp success rate reported in [4]. However, the planning success rate is 13.8% lower; the two primary reasons are user error and the design of the sensor mounts. Because the distance between the arm and the sensor mounting bars was small, the arm’s workspace was reduced and if the scooter was not positioned directly in front the object at the correct distance, planning would fail. Out of the 48 successful tasks, the average time to complete each task was 69s, including failed attempts. Successful attempts completed in an average of 50.7s. The average time of 69s for each task is a significant improvement (46%) over the time (128s) reported in [4].
C. Evaluating Pick And Place In-Situ

1) Experimental Protocol: Finally, we tested the pick-and-place functionality in a more complex version of the kitchen environment from the previous experiment. Along with the shelf and two tables, there was also a 108 cm tall high table, a 87 cm tall kitchen counter, and a ground area. On each of these seven surfaces we put four numbered tags 7 cm away from each other. We performed three trials. In each trial, 28 objects were randomly sampled from the object set, labeled from 1 to 28, and placed on top of the corresponding tag. The orientation of the objects was determined by a fair coin flip (1=upright, 0=not upright). Fig. 4b shows a setup of one trial. Each object was randomly assigned a target surface and the order the objects were to be pick-and-placed in was determined by randomly generating a permutation of 1 to 28. These trials were run with manual grasp selection in pick-and-place mode by an expert user. At the start of each task, the scooter was driven to Start Point 2 (seen in Fig. 4a) and then up to the next object to be grasped. After the object was grasped, the scooter was driven to the target surface and the object was placed on top of it. During the experiment, the user was allowed to perform pick-and-place actions to separate the target object from nearby objects.

We evaluated the same metrics as in Section VII-B. In addition, place detection success was defined to mean that the target place point indicated by the laser was detected successfully. Place success was defined as when the system placed the object qualitatively near the pointed-at location.

2) Results: Out of the 84 tasks (3 trials with 28 tasks each trial), 81 succeeded giving us a 96.4% task success rate. In 13 of the successful tasks, it was necessary to move a nearby object out of the way (we did this by performing a separate pick/place with the scooter) in order to reach the target object. As is shown in Table II during the 81 successful tasks, 135 out of 150 grasp detections succeeded, 95 out of 108 grasping attempts succeeded, and 92 out of 95 placing attempts succeeded. The planning success rate in this experiment was only 85.3% (203 out of 238 planning attempts for both grasping and placing) due to the same user and sensor mount design errors stated in VII-B. In pick detection failures, the most significant failure mode (12 out of 15) was that there was no grasp on the target object. That is mostly because the IK solver failed to generate solutions for those grasps due to the same reason as the planning failures. The average time needed for each pick-and-place task was 130s excluding time for driving. Note that this is higher than the sum of grasping total and placing total in Table II because it was not guaranteed that each task could be finished by only one grasping and placing attempt.

VIII. DISCUSSION AND LIMITATIONS

Overall, the system showed reliable functionality for both pick-and-drop and pick-and-place in an open world environment, suggesting that we may soon be able to begin user studies with the target populations. However, there are several limitations which may impact how successful these studies will be. The most significant limitation is the low planning success rate in open world environments. In the current state, the user might have to reposition the scooter multiple times in order to grasp the desired object. One way to improve this would be to increase the distance between the arm and the sensor mounts, thereby increasing the arm's workspace. The second major limitation is the effect of direct sunlight on the system which results in poor point cloud data and laser pointer detection. While dealing with poor point cloud data is quite challenging, we could achieve a higher laser detection success rate by improving the detection algorithm to be robust to reflected sunlight or by using a different depth sensor (e.g., Intel RealSense).

Although user studies of the HRI are still future work, we can identify a few limitations in the current design. Primarily, the system requires the user to have fine upper-limb motor control, making it difficult for some people with disabilities to use. In the future, we plan to implement several different user interfaces for different abilities (e.g., voice control or sip and puff control of a servo mounted laser pointer device). In addition, we plan to create another platform with a powered wheelchair, which will be able to navigate autonomously along with the current manipulation functionality, allowing the system to move, if needed, to pick up the desired object.

| Table II | RESULTS FOR PICK AND PLACE IN-SITU |
|----------|-----------------------------------|
|          | Success Rate | Average Time |
|          | Detection | Planning | Execution | Point Cloud | Laser | GPD | Grasp Filter | Grasp Selection | Planning | Execution | Total |
| Grasping | 90% (135/150) | 80% (108/135) | 88% (95/108) | 8.3s | 6.0s | 3.7s | 4.9s | 2.5s | 7.5s | 17.1s | 53.3s |
| Placing  | 99% (103/104) | 92.2% (95/103) | 96.8% (92/95) | 9.5s | 4.7s | - | - | - | 1.1s | 15.6s | 35.1s |

Fig. 5. System grasping an object in-situ
REFERENCES

[1] L. Kraus, E. Lauer, R. Coleman, and A. Houtenville, “Disability statistics annual report,” University of New Hampshire: Durham, NH, USA, 2018.

[2] C. Martens, N. Ruchel, O. Lang, O. Ivlev, and A. Graser, “A friend for assisting handicapped people,” IEEE Robotics Automation Magazine, vol. 8, no. 1, pp. 57–65, March 2001.

[3] A. Jain and C. C. Kemp, “El-e: an assistive mobile manipulator that autonomously fetches objects from flat surfaces,” Autonomous Robots, vol. 28, no. 1, p. 45, 2010.

[4] M. Gualtieri, J. Kuczyński, A. M. Shultz, A. T. Pas, R. Platt, and H. Yanco, “Open world assistive grasping using laser selection,” in 2017 IEEE International Conference on Robotics and Automation (ICRA), May 2017, pp. 4052–4057.

[5] K. Tsui, D. Kim, A. Behal, D. Kontak, and H. Yanco, “I want that”: Human-in-the-loop control of a wheelchair-mounted robotic arm,” Applied Bionics and Biomechanics, vol. 8, no. 1, pp. 127–147, 2011.

[6] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, “Grasp pose detection in point clouds,” The International Journal of Robotics Research, vol. 36, no. 13-14, pp. 1455–1473, 2017. [Online]. Available: https://doi.org/10.1177/0278364917735594

[7] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt, “High precision grasp pose detection in dense clutter,” in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Oct 2016, pp. 598–605.

[8] C. C. Kemp, C. D. Anderson, H. Nguyen, A. J. Trevor, and Z. Xu, “A point-and-click interface for the real world: Laser designation of objects for mobile manipulation,” in 2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI), March 2008, pp. 241–248.

[9] Y. S. Choi, C. D. Anderson, J. D. Glass, and C. C. Kemp, “Laser pointers and a touch screen: Intuitive interfaces for autonomous mobile manipulation for the motor impaired,” in Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility, ser. Assets ’08. New York, NY, USA: ACM, 2008, pp. 225–232. [Online]. Available: http://doi.acm.org/10.1145/1414471.1414512

[10] F. Achic, J. Montero, C. Penaloza, and F. Cuellar, “Hybrid bci system to operate an electric wheelchair and a robotic arm for navigation and manipulation tasks,” in 2016 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO), July 2016, pp. 249–254.

[11] P. M. Grice and C. C. Kemp, “Assistive mobile manipulation: Designing for operators with motor impairments,” in RSS 2016 Workshop on Socially and Physically Assistive Robotics for Humanity, 2016.

[12] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” arXiv preprint arXiv:1605.02199, 2016.

[13] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, “Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics,” arXiv preprint arXiv:1703.09372, 2017.

[14] M. Kopicki, R. Detry, M. Adjigble, R. Stolkin, A. Leonardis, and J. L. Wyatt, “One-shot learning and generation of dexterous grasps for novel objects,” The International Journal of Robotics Research, vol. 35, no. 8, pp. 959–976, 2016.

[15] U. Klapk, D. Pangercic, R. B. Rusu, and M. Beetz, “Real-time cad model matching for mobile manipulation and grasping,” in Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on. IEEE, 2009, pp. 290–296.

[16] U. Vierack, A. Pas, K. Saenko, and R. Platt, “Learning a visuomotor controller for real world robotic grasping using simulated depth images,” in Conference on Robot Learning (CoRL), 2017.

[17] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in ICRA workshop on open source software, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[18] L. Rokach and O. Maimon, “Clustering methods,” in Data mining and knowledge discovery handbook. Springer, 2005, pp. 321–352.

[19] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[20] R. B. Rusu, “Semantic 3d object maps for everyday manipulation in human living environments,” Ph.D. dissertation, Computer Science department, Technische Universitaet Muenchen, Germany, October 2009.

[21] K. P. Hawkins, “Analytic inverse kinematics for the universal robots ur-5/ur-10 arms,” Georgia Institute of Technology, Tech. Rep., 2013.

[22] R. Diankov and J. Kuffner, “Openrave: A planning architecture for autonomous robotics,” Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34, vol. 79, 2008.

[23] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, “Finding locally optimal, collision-free trajectories with sequential convex optimization.” in Robotics: science and systems, vol. 9, no. 1. Citeseer, 2013, pp. 1–10.