2017 13th IEEE Conference on Automation Science and Engineering (CASE)
Xi'an, China, August 20-23, 2017

Robust Shared Autonomy for Mobile Manipulation with Continuous Scene Monitoring

Wolfgang Merkt[1], Yiming Yang[1], Theodoros Stouraitis[1], Christopher E. Mower[1], Maurice Fallon[2], Sethu Vijayakumar[1]

***Abstract*— This work presents a fully integrated system for reliable grasping and manipulation using dense visual mapping, collision-free motion planning, and shared autonomy. The particular motion sequences are composed automatically based on high-level objectives provided by a human operator, with continuous scene monitoring during execution automatically detecting and adapting to dynamic changes of the environment. The system is able to automatically recover from a variety of disturbances and fall back to the operator if stuck or if it cannot otherwise guarantee safety. Furthermore, the operator can take control at any time and then resume autonomous operation. Our system is flexible to be adapted to new robotic systems, and we demonstrate our work on two real-world platforms – fixed and floating base – in shared workspace scenarios.**

To the best of our knowledge, this work is also the first to employ the *inverse Dynamic Reachability Map* for real-time, optimized mobile base positioning to maximize workspace manipulability reducing cycle time and increasing planning and autonomy robustness.

I. Introduction

The level of autonomy of a robot is directly correlated with the predictability of the task and environment. Robots executing a predictable task in a foreseeable environment such as in a factory setting work fully autonomous, while for field robots where variance is high, teleoperation is still the accepted gold standard with the human as the decision maker and the robot acting as an extension of the human operator. Many applications in industry are repetitive and require high levels of concentration and manual dexterity, often coupled with the human operator solely performing scene monitoring as well as hazard detection and prevention. As such, the operator is prone to fatigue that has been linked to serious accidents in the past. As a result, research has investigated guided teleoperation with on-the-fly synthesized constraints to provide assistance or resistance via force feedback – for instance in surgical robotics, e.g. via virtual fixtures. This guidance and protection of sensitive areas leads to a reduction in the concentration devoted to the task as it relaxes the mental criteria for task success and failure [1].

Different state-of-the-art teleoperation and shared autonomy approaches have been demonstrated at the DARPA Robotics Challenge (DRC) Finals in July 2015 for controlled, mostly static environments (*"high-level repeatable and low-level adaptable"*, [2]). However, more or fully autonomous systems are challenging in dynamic and unpredictable environments with clutter due to the sheer complexity of dealing with rare events. Shared autonomy is often perceived as a middle ground, combining autonomous sequences by the robot with input from a human operator for high-level decision making reducing cognitive load and leading to more reliable systems.

Fig. 1. The bimanual mobile manipulator executing a shared workspace task: model-free segmentation of target objects with automatic mobile base placement selection and navigation to place them in a bin. The system automatically adapts to various dynamic changes, including changing object and target locations during execution as well as scene monitoring for safe avoidance and replanning of motion to accommodate human operators.

These systems are especially important for manipulation and exploration in hazardous applications, such as for space exploration or disaster recovery with high-latency, low bandwidth communication and intermittent transmission cut-outs making teleoperation impractical. Furthermore, we argue that shared autonomy is key to complement high-level human direction with high-frequency, closed-loop dexterity of a robotic system.

A. Background

We refer to *teleoperation* as a control method of a robot that directly maps operator input to robot actions. An *autonomous* robot is an agent that perceives its environment, forms decisions based on its perception of the world, and realizes actions according to these decisions. We define *shared autonomy*, similar to Sheridan in [3], to be a blend of teleoperation and autonomy that realizes robot actions.

*This research is supported by the Engineering and Physical Sciences Research Council (EPSRC, grant reference EP/L016834/1). The work has been performed at the University of Edinburgh under the Centre for Doctoral Training in Robotics and Autonomous Systems program.

[1]School of Informatics, The University of Edinburgh (Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, United Kingdom). Email: wolfgang.merkt@ed.ac.uk.

[2]Oxford Robotics Institute, University of Oxford (Felstead House, 23 Banbury Road, Oxford, OX2 6NN, United Kingdom).

Several applications for teleoperation using immersive virtual reality head-mounted displays and remote controllers with haptic feedback have been demonstrated in recent years [4], [5], e.g. with the PR2[1] and Baxter[2] robots. While the former used low-cost, consumer-grade hardware such as the Oculus Rift, the SRI Taurus Dexterous Robot[3] is an example of military-grade, immersive teleoperation equipment for Explosive Ordnance Disposal (EOD) tasks. Fok et al. [6] successfully demonstrated the feasibility of web-based teleoperation by having inexperienced human operators interface with a humanoid robot through a web browser on a smartphone to complete a bimanual telemanipulation task.

Extensive studies have been performed to deduce the key aspects of robot control methods that must be developed towards successful shared autonomy systems. The DRC has been the largest demonstration of the state-of-the-art in disaster response robotics, where most systems implemented some form of shared autonomy. Yanco et al. [7] identified the desired attributes of a shared system and highlighted specific design guidelines for being successful at the DRC, arguing that the design of a user interface is of high importance in regards to the overall success of a task.

The DIRECTOR interface and system architecture used to control the ATLAS robot, developed by Team MIT, is described in [8] and [9] respectively. It features a shared autonomy system backed by interactive, assisted perception, and trajectory optimization-based motion planning [10] where task sequences can be created from high-level motion primitives and constraints. The operator (or multiple operators) maintains a supervisory role, pausing execution and adjusting affordances and constraints in response to changes.

A similar approach has been taken by Team ViGIR which fitted affordance templates with semantic actions [11], [12]. They also used a virtual robot model for planning and review in a supervised semi-autonomous approach.

The JPL Team details their hardware design, algorithms, and system for ROBOSIMIAN in [2] and [13]. Their semi-autonomous system utilizes a behavior planner and stored motion primitives along with nonlinear trajectory optimization and review and approval of candidate plans by a human supervisor, with the operator also assisting in object fitting.

Similar semi- and shared autonomy systems have been developed to carry out related tasks. Stückler et al. [14] use motion key frames generated by an operator based on segmented perception data with the autonomous system interpolating between key frames given constraints. Walter et al. [15] describe an autonomous forklift able to operate safely in a shared workspace with humans accepting task-level commands through natural language and pen-based gestures. A high level control method implementing shared autonomy for debris clearance is presented by Kaiser et al. [16] where the system proposes affordances and actions for the operator to select and command.

[1]Cf. http://ros.org/news/2013/09/pr2-surrogate.html

[2]Cf. https://www.youtube.com/watch?v=JHIz-Y5qCmY

[3]Cf. https://www.sri.com/engage/products-solutions/taurus-dexterous-robot

B. Challenges

Key challenges for autonomous operation in unstructured real world environments include:

- *Perception:* Segmenting objects from single-view 2.5D data often fails in real-world settings due to misalignments and the limited area observed. This is exacerbated when the target affordance is occluded or part of the robot geometry further obstructs the view.
- *Robust and fast motion planning:* Synthesizing collision-free motion which handles the robot's redundancy (or lack thereof) to solve a task as well as work with real sensor data.
- *Dynamic changes:* Causing created plans or motion under execution to be in collision, requiring adaptive change or soft stopping and replanning.
- *Environment representation and collision checking:* Artifacts in real sensor data can lead to spurious planning failures, with approaches often resorting to disabling collision checking and having a human do the verification of the final trajectory.
- *Computation, memory, and communication:* Real-world scenarios impose constraints from available on-board computing power and memory with online planning capability and feature communication constraints to a remote operator, resulting in a trade-off of between algorithm and solution quality and execution speed.

C. Overview

In this work we present a fully integrated shared autonomy system with environment mapping, autonomous stance selection and navigation, model-free object segmentation, automatic grasp affordance selection, collision-free motion planning and execution, and continuous, dynamic scene monitoring and failure recovery (Section II).

In particular we expand on prior work on shared autonomy [8] by

1) Incorporating *dense visual mapping* to capture and fuse multiple views and sensors into a dense, 3D representation of the workspace to increase robustness of model-free affordance segmentation algorithms.
2) Integrating scalable and robust *collision-free motion planning* using our proposed efficient, hybrid scene representation.
3) Adding environment awareness and adaptation to dynamic changes through *continuous scene monitoring* with failure recovery enabling safe operation in shared workspaces.
4) Selecting an *optimized base position* for floating-base robots to maximize manipulability and decrease cycle times; the first employment of *inverse Dynamic Reachability Maps* [17] to improve the robustness of autonomous motion planning and execution.

We demonstrate the flexibility and adaptability of our system in experiments with an industrial manipulator and a mobile bimanual robot (Section III, Figure 1).

II. System Architecture

In this section we describe the core components of our integrated shared autonomy system: perception (Section II-A), continuous scene monitoring (Section II-B), a shared autonomy user interface (Section II-C), and collision-free motion planning (Section II-D). We additionally detail extensions required to apply this framework to a mobile robot (Section II-E). The full system is shown in Figure 2.

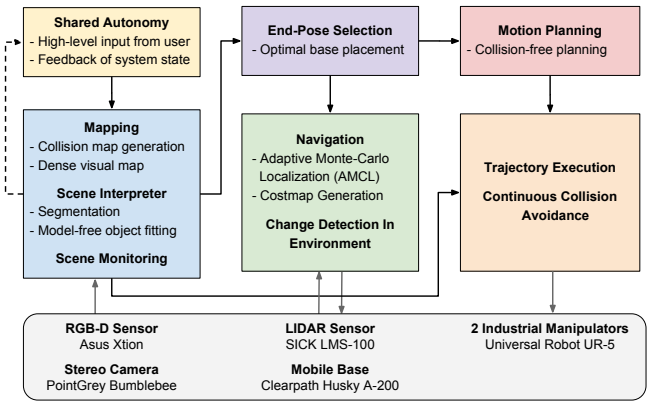


Fig. 2. Overview of the implemented system.

A. Perception

At the core of our perception subsystem is a continuously updated map encoding free, occupied, and unknown space in an accurate, dense yet memory-efficient representation of the environment.

1) Filtering: In order to reduce artifacts which hinder the reliability of motion planning and object segmentation algorithms, raw RGB-D and block-matched stereo sensor data is filtered in a preprocessing step to remove self-observations, shadow points and noise. Multiple pointcloud sources can be adjoined and fused. The pre-processing pipeline is shown in Figure 3. In a first step, a pass-through and downsampling filter is applied into the sensor z-direction to cut out noisy areas which are far away (beyond the area of interest) or too close for the sensor to provide reliable readings as well as to regularize samples onto a voxelgrid. It also removes invalid points and mixed pixels. In a second step, the pointcloud is projected into the robot base frame and measurements outside the work area are filtered, before self-observations are removed and unconnected patches eliminated via a statistical outlier removal. Measurements from multiple pointcloud sensors, here a stereo camera and a RGB-D sensor, are fused.

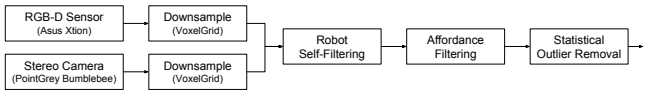
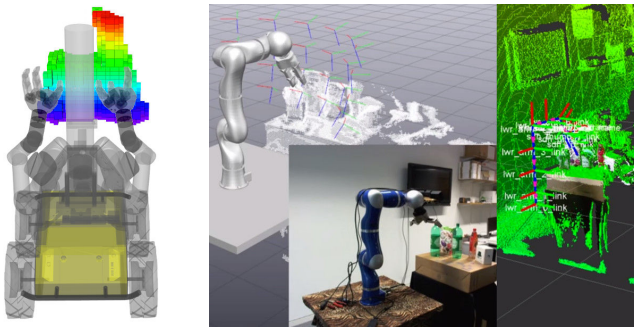


Fig. 3. Filter chain from raw RGB-D and blockmatched stereo data to data ready for insertion into the map.



(a) (b)

Fig. 4. Scene representations during the experiments: (a) Bimanual mobile manipulator with octomap used for planning, (b) Dual representation for fixed-base manipulator: Dense, visual map for segmentation shown on the left, and octomap for planning to the right.

2) Hybrid Environment Representation: The preprocessed and adjoined pointcloud is fused into a memory-efficient, probabilistic octree-based scene representation [18] that allows for change detection. Where available, we additionally fuse information from force/torque or fiducial sensing into the same environment map for collision detection.

In order to make use of multiple views to increase object segmentation and grasp affordance robustness, a dense, high resolution map based on the *Truncated Signed Distance Function* is created [19], an example of which is shown in Figure 10a. In the example of a highly repeatable and accurate fixed base manipulator with end-effector mounted pointcloud sensor, we replaced visual tracking with forward kinematics to increase mapping speed and reduce cycle times.

This dual environment representation (Figure 4) allows efficient collision avoidance, change detection and tracking while providing a high fidelity fused map for accurate model-free affordance segmentation.

3) Segmentation: Crucially, the perception system needs to be able to extract objects of interest based on high-level user input and without prior models. These affordances will be segmented from the higher resolution representation resulting in a more accurate modeling of the areas of interest. This produces an efficient, hybrid representation of segmented affordances and a dense, discretized environment occupancy map for planning and autonomy.

In order to segment objects of interest without a prior model, we use a combination of geometric insights and task-informed assumptions in an algorithm similar to [20]. First, we assume that objects of interests are placed on an approximately horizontal surface and have sufficient clearance from one another for a gripper to pass between to pick them up. This allows us to use segmentation by normals to extract a large, continuous plane (e.g. table/shelf) and then Euclidean clustering to extract clusters distinct of individual objects. We then reconstruct a mesh through triangulation and fit an approximate bounding box shape primitive affordance annotated with candidate grasps to the mesh. The whole process can be seen in Figure 10 in Section III-A.

B. Continuous Scene Monitoring

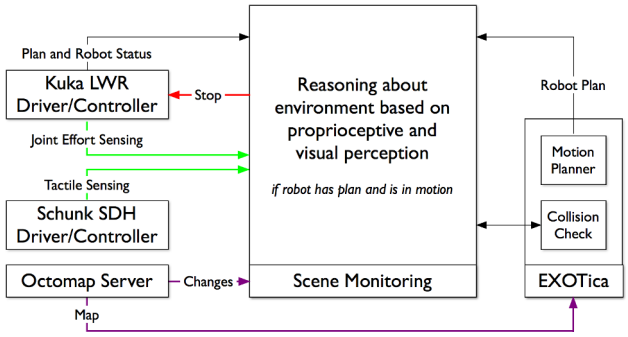


Fig. 5. The implemented scene monitoring utilizing different perception modalities: visual RGB-D data, tactile, and joint effort sensing. Diagram showing modules active during Experiment 1 (Kuka LWR).

Key to safe operation in shared workspaces is the ability to identify whether dynamic changes affect the intended robot motion. We hereby distinguish between changes that alter targets and affordances triggering replanning as well as updates which affect the safe and collision-free execution of the motion.

The continuously integrated discretized occupancy map is the basis for tracking changes. Our scene monitoring and reasoning (shown in Figure 5) hereby works similar to the one described in [21]. Instead of analyzing a sequence of swept volumes, we check the key poses of the trajectory currently being executed directly against the collision map and changed areas upon every map update. This choice enables us to efficiently run scene monitoring collision queries on CPU on-board at sensor frame rate, with the GPU free to be used e.g. for dense visual mapping. If a collision of a future key pose with the map update is detected, we halt the execution and fall back to the human operator. In the meantime, the scene continues to be monitored and motion execution is resumed once the trajectory would be collision-free again. Alternatively, a replanning is triggered on expiration of a countdown.

C. Shared Autonomy

A shared autonomy user interface serves as an abstraction layer above task complexity and allows the user to provide high-level objectives, gives feedback as well as involves the operator in decision-making if necessary.

Our shared autonomy builds on the task execution system described in [8] which fills a sequence of task primitives with details acquired through operator-assisted perception online. The operator can review, pause, and amend the execution at any time, and the autonomous mode can be resumed immediately after a phase of manual operation.

We expand on [8] to extend the task sequence system with automatic review and approval of planned motions through continuous scene monitoring and reasoning to reduce the amount of human intervention required, and only fall back to the operator when absolutely necessary. In order to achieve this, a task tree with task dependencies is defined that is automatically expanded to synthesize sequences comprised of high-level action primitives in response to perception input and changes in the environment.

Each high level task primitive is automatically expanded into a series of verifiable low-level tasks, and the success criteria associated with each action are monitored. For instance, grasping is executed as a power grasp with force, tactile, or visual feedback serving as a success/completion criterion with visual inspection for recovery. Automated reasoning immediately starts execution of valid trajectories to reduce cycle time.

D. Collision-free Motion Planning

Our system uses a combination of sampling- and optimization-based planning algorithms from the Extensible Optimizaton Toolkit (*EXOTica* [22]) to synthesize collision-free manipulation trajectories. Using affordances and occupancy grids from our perception module (Section II-A, Figure 4), motion planning problems are centered around constraint sets [23] that can be composed to represent high-level objectives. Given an environment and reachability and manipulation constraints of an affordance, an optimization-based inverse kinematics solver [10] computes a goal configuration $\mathbf{q}_{goal}$. Given start and goal configurations $\mathbf{q}_{start}, \mathbf{q}_{goal}$, RRT-Connect is used to find a trajectory [24], [25].

E. Extension to Floating-Base Systems

The system discussed so far is generic and was experimentally applied to and validated with a fixed-based manipulator. In the following paragraphs, we will highlight additional components required to extend this system to mobile, floating-base robots.

The kinematic reachability of low-cost, non-redundant mobile platforms is often limited by potential self-collisions, mounting points, and complex environments (cf. Figure 8 for a reachability map similar to [26] showing the manipulability of our mobile manipulator platform).

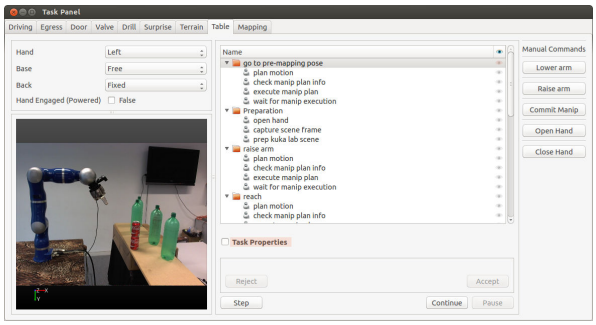


Fig. 6. The shared autonomy task panel view for the table clearing task. A third-person surveillance camera view is shown alongside with the expanded task tree. Manual intervention actions are accessible via buttons on the right panel. The progress through the task tree is visualized at every stage and the operator can pause and manually step through execution, as well as resume autonomous operation. A confirmation dialog is presented if the system falls back to the operator for approval if it is not confident it can proceed safely.

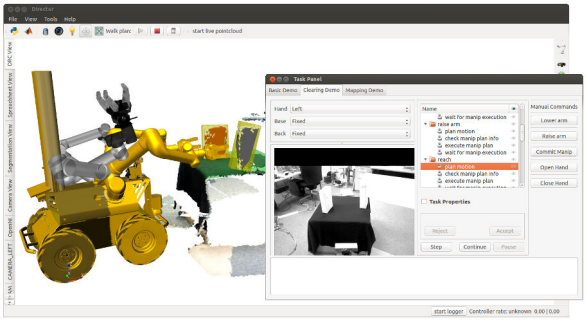
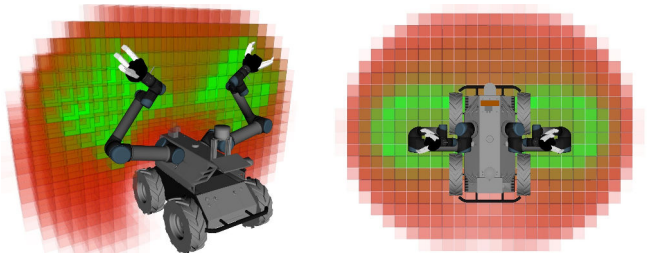


Fig. 7. The user interface for the bimanual mobile manipulator showing live perception data, segmented objects and fitted affordances as well as candidate plans (gold). Situational awareness camera data is shown in the shared autonomy panel which automatically executes the candidate trajectory if deemed safe by the continuous scene monitoring. The operator can manually pause and step as well as adjust affordances and plans in the interactive user interface.



(a) Backview (b) Topdown view

Fig. 8. Reachability map for the dual-arm mobile manipulator used during our experiments. The mounting point and non-redundancy of the individual arms severely restricts workspace manipulability, while the compact construction means that sensor placement and highly manipulable workspace are not always aligned highlighting the usefulness and need for intelligent mobile base positioning to maximize task success.

1) Optimized Floating Base Positioning: In order to increase manipulability, we leverage the mobile base to reposition the manipulator based on intended motion plans. We select an appropriate standing pose to maximize manipulability using *inverse Dynamic Reachability Maps* (iDRM), the first time it is being applied to improve autonomous operation, reduce cycle time, and increase robustness. Maximizing manipulability is important to cope with dynamic change and perception inaccuracies to an extent as it maximizes the likelihood that nearby task space areas can be reached with the same base placement.

We recap the core principles of the iDRM algorithm as described in [17]. During an offline preprocessing step, a large number (millions) of self-collision-free robot configurations are generated and transformed such that the end-effector is at the origin of a voxelgrid. Configurations which can reach a voxel with the end-effector at the origin are stored in the reach list of the corresponding voxel. Unlike previous algorithms for floating base placement, iDRM stores voxels occupied by configurations in an occupation list during this offline step such that only a single collision query is required to find the voxels which are occupied in the environment representation and filter associated configurations. The remaining subset of the iDRM is collision-free and the most suitable end-pose for the task, i.e. a collision-free base location from where the robot can achieve the desired end-effector pose with highest manipulability, can be selected based on a pre-calculated manipulability score. It was shown in [17] that iDRM is able to find valid end-poses in real-time in complex environments, which can then bootstrap bidirectional motion planning algorithms. Thus, the end-pose information, which is normally provided by a human operator, is crucial for robot autonomy by reducing planning failures and increasing the overall success rate.

2) Navigation: During mobile manipulator experiments, we used a front-mounted horizontal laser rangefinder for navigation and localization against a static map (Adaptive Monte-Carlo Localization [27]).

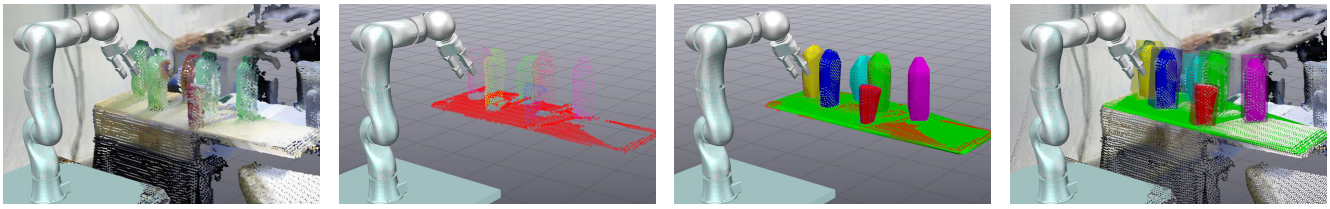
Goal base positions $\mathbf{p}_{goal}$ are computed using iDRM and passed to the ROS navigation stack which provides cost map generation and path planning out-of-the-box.

III. System Evaluation

We validated the flexibility and adaptability of our system with hardware experiments on two different platforms. First, we use a 7-DoF Kuka LWR3+ manipulator with a Schunk SDH dexterous hand to clear a table. For perception, an Asus Xtion Pro Live RGB-D sensor is mounted on the end-effector for dense visual mapping and continuous scene monitoring. Second, we use a bimanual Clearpath Husky with two 6-DoF Universal Robot UR5 manipulators fitted with Robotiq 3-finger grippers to clear a scene. For this system, the perception is based on an identical Asus Xtion, a PointGrey Bumblebee2 stereo camera, and a Sick LMS-100 LIDAR sensor. All sensors are mounted on-board the respective platform; no external sensing/perception is used.

For the industrial manipulator, computation is carried out on a personal computer with an Intel Core i7-4770 @ 3.4 GHz CPU, 16GB 1600 MHz DDR3 memory, and a Nvidia GeForce GT645 GPU. For the experiments with the bimanual mobile manipulator, most computation is performed on-board the robot (Intel Core i5-4570TE @ 2.7 GHz with 8GB 1600 MHz DDR3 RAM), with mapping, planning, and shared autonomy offloaded to the operator workstation (same as above). Inter-process communication is maintained using ROS and LCM [28], with on-board compression allowing the operation via a wireless link without line of sight. Thereby, peak bandwidth use for streaming two depth and three situational awareness camera feeds beyond telemetry and lidar is approximately 3 MB/s. This allows for experiments with a blend of teleoperated and autonomous sequences with no line of sight as depicted in Figure 9.

In our experiments, we illustrate that by using the same system architecture we are able to generalize the manipulation and continuous scene monitoring skills of a fixed base robot to a floating base system with two arms. As it is evident, the mobile robot requires additional components compared with the fixed-base manipulator, which allow us to take full advantage of its advanced capabilities.



(a) Map (b) Clusters (c) Meshes (d) Affordances in Map

Fig. 10. Steps of our model-free affordance segmentation pipeline: Starting with a dense visual map, a table surface and clusters distinct point clusters are extracted, and meshes fit via triangulation. Finally, primitive shape affordance are fit to the meshes and annotated with possible grasps.

A. Experiment 1: Table Clearance with a Kuka LWR arm

The first experiment is an autonomous table clearance task wherein the robot proceeds to identify objects on a surface given a single point on that surface and synthesize a plan for clearing the table. A dense visual map is created through volumetric fusion and objects and affordances are segmented (Figure 10), with the corresponding occupancy grid for collision-free motion planning shown in Figure 4b. The shared autonomy interface is shown in Figure 6. In this set of experiments, tactile information from the gripper is used during grasping and joint effort sensing used as part of the continuous scene monitoring, cf. Figure 5.

B. Experiment 2: Autonomous Scene Clearance with a bimanual Clearpath Husky

In the second experiment, a mobile manipulator picks up objects in a shared workspace with humans and places them into a garbage bin (Figure 1). The input from the operator is limited to providing a single point in the scene denoting the surface for the robot to clear. This experiment is verified with two different scenarios, where the robot uses either one or both arms depending on the number of objects placed on the designated surface. Each of the two scenarios have been tested with more than 20 different variations of the environment, including different objects, tables, and configurations of the furniture. Samples of these executions are shown in the accompanying video (`https://youtu.be/5jFU7oCP4vk`). A striking feature of the proposed shared autonomy system is its capability to automatically determine the number of end-effectors required to efficiently pick the objects from the surface. Furthermore, in the case of multiple objects, it automatically determines whether both are reachable simultaneously through optimized base positioning using iDRM, and it leverages the floating base to reduce cycle time.

In the following, we detail the tasks during operation and how they are divided between the robot and the operator. It is worth noting that operator tasks only convey high level goals and confirmations to the robot.

1) *Operator* provides a point on surface to be cleared.
2) *Perception* segments the scene and identifies the number of objects on the surface.
3) *iDRM* computes an end-pose to grasp the objects.
4) *Robot* navigates to the base pose provided by iDRM.
5) *Operator* verifies that the robot has navigated to the area within the vicinity of the target surface and confirms the surface by again providing a point.
6) *Perception* re-segments the scene in order to ensure that its plans remain compatible with any changes.
7) *Robot* plans and executes arm motions to grasp the object in three substeps, first to reach pre-grasp frame, then to grasp frame and finally grasps the object.
8) *Operator* verifies that the object(s) has been grasped.
9) *Robot* moves its arms in driving configuration while searching for the bin and navigates to it.
10) *Robot* drops the objects in the bin.

1) Scenario 1: Single object on surface: In this scenario, we show the robot removing an object from the indicated surface. Once the target surface has been provided by the

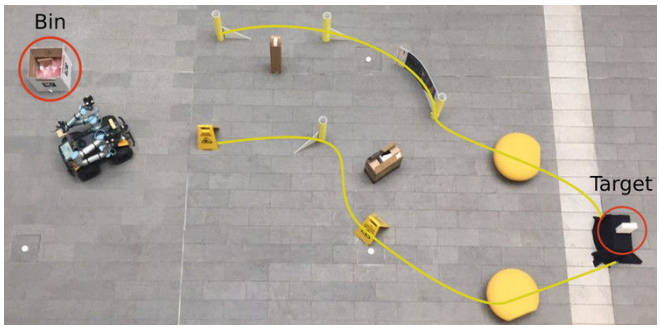


Fig. 9. Navigation and recovery task with no line-of-sight and restricted communication with teleoperated as well as autonomous sequences.

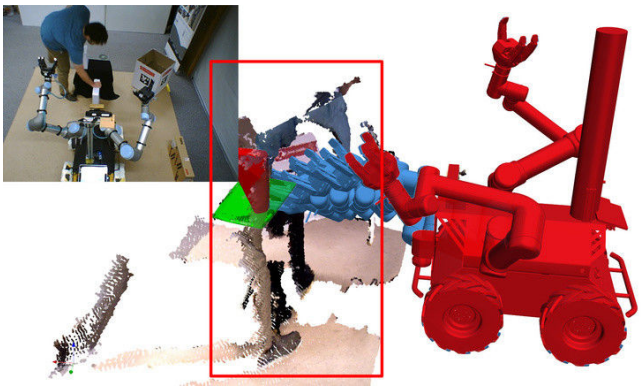
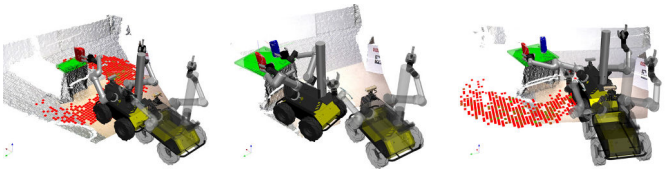


Fig. 11. The scene monitoring continuously integrates fused and filtered sensor data in an OctoMap and reasons about changes: Here, a human reaches into the robot workspace crossing the planned trajectory (shown in blue), and the robot halts execution (robot state shown in red).



(a) (b) (c)

Fig. 12. (a) Single arm end-pose computed by iDRM. The red and green squares illustrate the variety of different base poses that satisfy the six-DoF constraint to reach the pre-grasp end-effector frame. (b) Dual arm end-pose computed by iDRM. In this scenario there are limited feasible base poses, hidden under the robot, that satisfy the two six-DoF constraints, one for each arm. (c) Single arm end-pose computed by iDRM reaching only the red object. Note that given this pose the blue object is not reachable by the right arm.

operator, the robot automatically segments the scene and identifies that only one object is placed on the surface. Thus, the iDRM module provides whole-body end-pose solution using only the left arm, as shown in Figure 12a.

In the accompanying video, we clearly demonstrate that the robot is also able to cope with dynamic changes of the scene such as relocation of the target object and the bin.

Further, our scene monitoring is demonstrated in the accompanying video and in Figure 11. During execution, the workspace is monitored and motions are paused as soon as future trajectory waypoints stop being collision-free. The map updates at 13 Hz at a 3cm resolution (i.e. at frame rate – the sensor data is being captured at 15 Hz) with the verification whether the future trajectory key waypoints are collision-free running consuming approximately 50ms. Note, that this speed and interactivity can be scaled with the operating velocity of the manipulators by decreasing workspace voxelgrid resolution and an increased safety margin/padding.

2) Scenario 2: Two objects on surface: In the second scenario, the robot removes two objects from the indicated surface. The robot identifies autonomously the number of objects on the surface and utilizes the iDRM module to obtain a solution which satisfies a whole-body end-pose, reaching both objects simultaneously, as shown in Figure 12b. On the other hand, as shown in Figure 12c, if one were to first obtain a solution for the red object and subsequently for the blue, the robot would have to re-navigate to a new base-pose to reach the blue object with the right arm.

Analysis of task-wise timing

In Figure 13a and Figure 13b, we provide a detailed timing in a cumulative fashion for each task described above during the single object and the bimanual scenario respectively. As it is evident from both figures, the interaction time of the operator, colored in red, is minimal during the successful completion of the experiments. In addition, it is worth noting that the duration of the planning steps is negligible and most of the time is spent on execution. Execution times are large, due to the very restrictive velocity limits we are using to align with safety standards when the workspace is shared between a robot and a human.

IV. Discussion

The proposed shared autonomy system has been demonstrated to complete tasks with minimal high-level user input operating autonomously for the majority of the execution. It is able to deal with dynamic changes, such as updates of the target affordance or bin location, as well as an obstacle or human entering its shared workspace by safely pausing, replanning, and resuming motion execution.

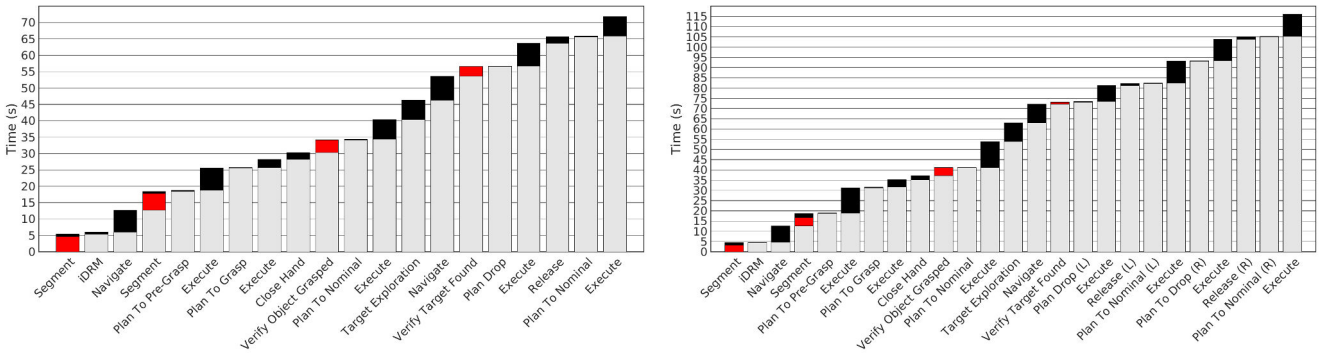
Our scene monitoring can be added on top of many motion planning and execution pipelines and runs at 20 Hz on CPU, which is responsive enough for operation on a moving platform. Work by Hermann et al. [21] checks swept volumes of trajectories on the GPU with additional predictive tracking of obstacles at 6-8 Hz and with replanning using a library of motion primitives. As a result, they interactively adapt the execution speed or interrupt motion if in collision. For our applications on mobile platforms where battery power is limited and a top-of-the-line GPU unlikely to be installed, the proposed scene monitoring is more applicable, yet pausing and replanning may result in short interruptions.

Selecting a suitable mobile base position improved autonomy robustness and adaptability to changes by maximizing manipulability. Especially in the second scenario, due to the appropriate placement of the base, both objects can be picked at once optimizing the execution time for any one reaching task. Inverse reachability maps have been previously used for instance during the DRC Trials [12]. Using the state-of-the-art iDRM algorithm which moves collision checking to the offline preprocessing phase, enabled real-time, interactive end pose queries during both teleoperation as well as increased robustness for our autonomous runs. However, the physical size of the workspace and robot model required large amounts of runtime memory to cover only a part of the workspace of the robot: for our single-arm experiments, we used 230k samples which translated to 520 MB memory, and the 1.3M samples of the bimanual dataset consumed 3.1 GB.

V. Conclusion

We have presented a robust shared autonomy system with continuous scene monitoring incorporating dense visual mapping, collision-free motion planning, and environment awareness. We furthermore present the first employment of iDRM [17] to improve the robustness of autonomous motion planning and execution by selecting a mobile base position which maximizes manipulability. The implemented continuous scene monitoring ensures safe execution of planned motion. It also paves the way for continuous adjustment of behavior according to changes in the environment, and enables fast and accurate manipulation allowing recovery when potential conflicts are detected during motion execution.

Future work includes continuous adaptation and local replanning of motion trajectories in response to environment changes captured by our scene monitoring, and to incorporate motion flow and predictive tracking similar to [21]. Additionally, we are interested in incorporating a novel improvement of dynamic reachability maps [29] leveraging hierarchies in the kinematic structure to reduce memory requirements



(a) Scenario 1 - Single object

(b) Scenario 2 - Two objects

Fig. 13. Time taken by individual steps during the experiments. Operator input is shown in red and the autonomous behavior is shown in black. Grey indicates the cumulative experiment time up to the start of the current step.

by several orders of magnitude allowing complete maps for reachability and motion planning to be stored in memory at runtime.

REFERENCES

[1] S. Park, R. D. Howe, and D. F. Torchiana, *Virtual Fixtures for Robotic Cardiac Surgery*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 1419–1420. [Online]. Available: http://dx.doi.org/10.1007/3-540-45468-3_252

[2] S. Karumanchi, K. Edelberg, I. Baldwin, J. Nash, J. Reid, C. Bergh, J. Leichty, K. Carpenter, M. Shekels, M. Gildner, D. Newill-Smith, J. Carlton, J. Koehler, T. Dobreva, M. Frost, P. Hebert, J. Borders, J. Ma, B. Douillard, P. Backes, B. Kennedy, B. Satzinger, C. Lau, K. Byl, K. Shankar, and J. Burdick, "Team RoboSimian: Semi-autonomous Mobile Manipulation at the 2015 DARPA Robotics Challenge Finals," *Journal of Field Robotics*, 2017.

[3] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. MIT Press, 1992.

[4] T. Rodehutskors, M. Schwarz, and S. Behnke, "Intuitive bimanual telemanipulation under communication restrictions by immersive 3D visualization and motion tracking," in *IEEE-Humanoids*, 2015.

[5] N. Britton, K. Yoshida, J. Walker, K. Nagatani, G. Taylor, and L. Dauphin, "Lunar micro rover design for exploration through virtual reality tele-operation," in *Field and Service Robotics*, 2015.

[6] C. L. Fok, F. Sun, M. Mangum, A. Mok, B. He, and L. Sentis, "Web Based Teleoperation of a Humanoid Robot," *arXiv preprint arXiv:1607.05402*, 2016.

[7] H. A. Yanco, A. Norton, W. Ober, D. Shane, A. Skinner, and J. Vice, "Analysis of Human-robot Interaction at the DARPA Robotics Challenge Trials," *Journal of Field Robotics*, 2015.

[8] P. Marion, R. Deits, A. Valenzuela, C. P. D'arpino, G. Izatt, L. Manuelli, M. Antone, H. Dai, T. Koolen, J. Carter, M. Fallon, S. Kuindersma, and R. Tedrake, "Director: a User Interface Designed for Robot Operation with Shared Autonomy," 2016.

[9] M. F. Fallon, S. Kuindersma, S. Karumanchi, M. E. Antone, T. Schneider, H. Dai, C. Prez-D'Arpino, R. Deits, M. DiCicco, D. Fourie, T. Koolen, P. Marion, M. Posa, A. Valenzuela, K.-T. Yu, J. A. Shah, K. Iagnemma, R. Tedrake, and S. J. Teller, "An Architecture for Online Affordance-based Perception and Whole-body Planning." *Journal of Field Robotics*, 2015.

[10] R. Tedrake and the Drake Development Team, "Drake: A planning, control, and analysis toolbox for nonlinear dynamical systems," 2016.

[11] A. Romay, S. Kohlbrecher, D. C. Conner, A. Stumpf, and O. von Stryk, "Template-based manipulation in unstructured environments for supervised semi-autonomous humanoid robots," in *IEEE-Humanoids*, 2014.

[12] S. Kohlbrecher, A. Romay, A. Stumpf, A. Gupta, O. Von Stryk, F. Bacim, D. A. Bowman, A. Goins, R. Balasubramanian, and D. C. Conner, "Human-robot Teaming for Rescue Missions: Team ViGIR's Approach to the 2013 DARPA Robotics Challenge Trials," *Journal of Field Robotics*, 2015.

[13] P. Hebert, M. Bajracharya, J. Ma, N. Hudson, A. Aydemir, J. Reid, C. Bergh, J. Borders, M. Frost, M. Hagman, J. Leichty, P. Backes, B. Kennedy, P. Karplus, B. Satzinger, K. Byl, K. Shankar, and J. Burdick, "Mobile Manipulation and Mobility as ManipulationDesign and Algorithms of RoboSimian," *Journal of Field Robotics*, 2015.

[14] J. Stückler, M. Schwarz, M. Schadler, A. Topalidou-Kyniazopoulou, and S. Behnke, "Nimbro explorer: semiautonomous exploration and mobile manipulation in rough terrain," *Journal of Field Robotics*, 2015.

[15] M. R. Walter, M. Antone, E. Chuangsuwanich, A. Correa, R. Davis, L. Fletcher, E. Frazzoli, Y. Friedman, J. Glass, J. P. How *et al.*, "A Situationally Aware Voice-commandable Robotic Forklift Working Alongside People in Unstructured Outdoor Environments," *Journal of Field Robotics*, 2015.

[16] P. Kaiser, D. Kanoulas, M. Grotz, L. Muratore, A. Rocchi, E. M. Hoffman, N. G. Tsagarakis, and T. Asfour, "An Affordance-Based Pilot Interface for High-Level Control of Humanoid Robots in Supervised Autonomy," *IEEE-Humanoids*, 2016.

[17] Y. Yang, V. Ivan, Z. Li, M. Fallon, and S. Vijayakumar, "iDRM: Humanoid Motion Planning with Real-Time End-Pose Selection in Complex Environments," in *IEEE-Humanoids*, 2016.

[18] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013.

[19] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald, "Kintinuous: Spatially extended kinectfusion," 2012.

[20] R. B. Rusu, N. Blodow, Z.-C. Marton, and M. Beetz, "Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments," in *IEEE-IROS*, 2009.

[21] A. Hermann, F. Mauch, K. Fischnaller, S. Klemm, A. Roennau, and R. Dillmann, "Anticipate your surroundings: Predictive collision detection between dynamic obstacles and planned robot trajectories on the GPU," in *IEEE-ECMR*, 2015.

[22] V. Ivan, Y. Yang, M. Camilleri, W. Merkt, and S. Vijayakumar, "EXOTica: a library for easy creation of tools for optimisation and planning," 2017.

[23] M. Fallon, S. Kuindersma, S. Karumanchi, M. Antone, T. Schneider, H. Dai, C. P. D'Arpino, R. Deits, M. DiCicco, D. Fourie *et al.*, "An Architecture for Online Affordance-based Perception and Whole-body Planning," *Journal of Field Robotics*, 2015.

[24] Y. Yang, V. Ivan, W. Merkt, and S. Vijayakumar, "Scaling Sampling-based Motion Planning to Humanoid Robots," in *IEEE-ROBIO*, 2016.

[25] J. J. Kuffner and S. M. LaValle, "RRT-Connect: An Efficient Approach to Single-Query Path Planning," in *IEEE ICRA*, 2000.

[26] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, "Reachability and capability analysis for manipulation tasks," in *First Iberian Robotics Conference*, 2014.

[27] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots," in *AAAI*, 1999.

[28] A. S. Huang, E. Olson, and D. C. Moore, "LCM: Lightweight Communications and Marshalling," in *IEEE-IROS*, 2010.

[29] Y. Yang, W. Merkt, V. Ivan, Z. Li, and S. Vijayakumar, "HDRM: A Resolution Complete Dynamic Roadmap for Real-time Motion Planning in Complex Environments," 2017, submitted to IEEE RA-L.