Vision-Based Robotic Pushing and Grasping for Stone Sample Collection under Computing Resource Constraints

Raphael Grimm, Markus Grotz, Simon Ottenhaus and Tamim Asfour

***Abstract*— Increasing the robustness of grasping actions and the recovery from failure is key to improving a robot's autonomy. Endowing robots with the ability to robustly grasp and manipulate unknown difficult objects such as stones is required for sample collection in unknown environments. In this paper, we present a complete system for robust grasping of stones, which integrates stone segmentation based on depth information, the generation of grasp hypotheses and pushing actions as well as their execution. In particular, our system has been designed to solve these tasks on robots with limited computing resources. We evaluate the performance in real robot experiments in the context of stone sample collection. The results show that such a challenging task is achievable under computing resource constraints.**

I. Introduction

To successfully interact with real-world scenarios, mobile robots must be able to cope with unstructured scenes containing unknown objects. One of the most fundamental tasks in such environments is to grasp and pick up an object in order to explore it or place it at a different location. Depending on the scene, even grasping an object can be a challenge. The complexity of the task results from the fact that the robot has no prior knowledge about the environment, its objects or their placement. Furthermore, grasping of objects may be unfeasible due to the way the objects are placed. In these cases rearranging the objects (e.g. by pushing actions) is a promising way to successfully facilitate grasping an object. This problem increases in difficulty if common assumptions such as regular object shapes, textures with well-defined features or high color differences within the scene do not apply. Approaches that use convolutional neural networks (CNN) have been successfully applied to perform grasping actions in similar scenarios. However, they require a large amount of computation resources. While this is acceptable for a stationary robot (e. g. in an industrial workcell), it is not appropriate for a mobile robot with limited computing resources that usually have to be shared with other tasks

In this paper, we consider such a scenario where the robot has to collect stone samples from a bed of gravel as shown in Fig. 1. These stones have irregular shapes and a color similar to the gravel. From robot perception perspective, the visual homogeneity of the unstructured scene increases the difficulty of the visual tasks (e. g. segmentation) required for detecting grasp hypotheses. It is difficult to track object hypotheses after

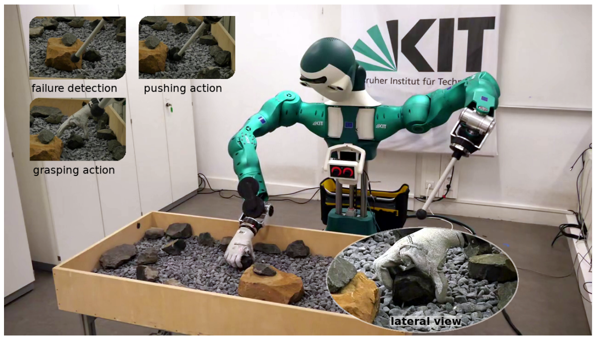

Fig. 1: ARMAR-6 autonomously collecting stone samples. An interactive perception strategy is employed to increase the robustness. To further improve the grasping success rate the motion of the hand is controlled to perform a sweeping motion.

interacting with the scene (e. g. by pushing actions) on the basis of image features, since no stable image features are available due to the small color differences in the scene. In such scenarios, pixel color provides only a low amount of information. Thus, it is not used by our approach, which relies only on depth data and interaction forces for grasping. Such a scenario can arise during autonomous exploration with robots in remote areas or during space missions.

Besides the successful collection of stone samples, further goals of our approach are a short runtime and low computational cost to make it suitable for small mobile robots with limited resources such as rovers for space exploration.

We present an interactive perception system for grasping loose stones in such environments, combining (1) the generation of grasp hypotheses and the selection of feasible actions for sample collection as well as (2) an interactive perception strategy that integrates manipulative capabilities with perception to verify the generated grasp hypotheses. The system requires only a low amount of computing resources while providing grasp hypotheses at 100 Hz. It is robust against noise in the input data by using interactive perception and spatio-temporal filtering of grasp hypotheses. We evaluate our approach in different robot experiments while comparing the runtime of different hardware setups.

II. Related Work

The survey papers [1] and [2] provide comprehensive overviews of approaches for grasp planning and synthesis. Following those, we group approaches for grasp planning into planning for (1) known objects, or (2) unknown objects. Methods for grasping known objects ([3], [4]) use a database, which stores object models with corresponding grasps. In

This work has been supported by the Helmholtz Association under the Helmholtz Future Project ARCHES (ZT0033).

The authors are with the High Performance Humanoid Technologies Lab, Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany. {raphael.grimm, asfour}@kit.edu

addition to this prior knowledge about the object, such approaches require object detection and pose estimation at runtime. Our presented approach requires neither a priori object knowledge nor object detection or pose estimation. In contrast, methods for grasping unknown objects extract geometric information based on perceptual data and use it to generate grasp hypothesis. Recently, machine learning has been successfully used in a variety of approaches to determine grasp hypotheses for unknown objects. The approaches [5] and [6] directly determine an optimal grasp hypothesis for a segmented target object. Similarly, our approach requires segmentation, but it can cope with imprecise or under segmentation by refining it through pushing actions. In addition, our approach generates multiple grasp hypotheses per object. Other approaches ([7], [8]) use unsegmented RGB-D images to estimate gripper poses suitable for grasping an object. Although color information can be valuable for grasping objects, e.g. in household scenarios, we cannot use this information, due to the mostly monochromatic scene. Since we have to be able to deal with homogeneously colored environments, we do not rely on any color information. Further approaches ([9], [10]) use only depth data and show that color information is not necessary for successful object grasping. The main difference to our approach is that they, like many other discussed approaches ([5]–[8]), use convolutional neural networks (CNN) to accomplish this task. CNNs have shown that they are able to generate grasp hypotheses, but the employed CNN architecture requires high computing resources. In contrast, our approach employs methods using only a low amount of computation resources. Furthermore these approaches focus on generating grasp hypotheses for the current scene and do not integrate information from previous time steps or information about the change in the scene as result of executed actions. Our approach integrates both to improve performance.

Approaches that consider manipulating the scene to facilitate a task are subsumed under the term *Interactive Perception (IP)*. The survey in [11] defines IP approaches, as approaches using sensor data from multiple time frames and utilizing *forceful* interactions with the environment (e.g. rearrangement of objects) to gain new information about the scene. The authors further categorize IP approaches depending on their application areas. Following this terminology, our approach classifies as IP for grasp planning, since we (1) rearrange the scene by pushing and (2) derive information from the results of executed grasping actions. Both are *forceful* interactions with the scene and are used to facilitate grasping. Other IP approaches ([12]–[14]) use pushing actions to facilitate grasping and learn a policy for selecting the next action to perform. The approaches in [12] and [14] also rely on color data that provides little information in the scenario considered in this paper. In [15], the authors propose a learning approach that uses unstacking actions to grasp a potentially completely occluded target object, but the approach requires object detection and that the class of the target object is known. All these approaches ([12]–[15]) successfully rely on CNNs and therefore require more computation resources than the proposed solution. The approaches in [16] and [17] successfully use pushing actions to improve object segmentation and do not rely on CNNs. However, both require color or image features that allow tracking between pushing actions, which is not necessary for our approach.

Overall, previous approaches either use color information or image features, require large computational resources, or do not make use of interactive perception, while in our scenario we have only access to depth information and limited computing resources for processing.

III. Approach

Our approach to address grasping and collecting stone samples from a gravel bed consists of three main parts: (1) Scene segmentation and generation of grasp hypotheses based on 3D point clouds, (2) collection of stone samples on the basis of the generated grasp hypotheses and (3) scene manipulation through pushing actions in cases where grasping is not possible. Since we focus on collecting stone samples that are mostly convex, we take advantage of this prior knowledge and design our approach to work well with convex objects. We generate grasp hypotheses on the basis of boxes that approximate the detected segments. Due to limited computing resources we have to use methods for scene segmentation and shape approximation that require a low amount of resources. These are more susceptible to noise than more sophisticated or machine learning based approaches. To cope with noise, we filter outliers at several stages: (1) We remove outliers from the input point cloud, (2) when generating the shape approximations and (3) from the resulting grasp hypotheses. Even then, the resulting grasp hypotheses do not always lead to a successful grasp execution. Our approach recognizes when execution fails and avoids executing similar grasps in the future. If stones are detected within the scene, but grasping is not possible, the robot uses pushing actions to improve segmentation and enable grasping. If the scene is larger than the camera's field of view, the approach can generate platform movements to cover the whole scene. This requires to have some method of self localization or odometry. These parts are implemented in the five-stage pipeline, which is executed in a loop until a termination criterion is reached (e. g. the scene is empty). The pipeline is shown in Fig. 2 and is described in more detail below.

A. *Visual Perception based on Depth Data*

Since color data contains little information for the given task, we rely only on depth information for segmentation and all further steps. Since scene segmentation is not the focus of this paper, we use methods available in the Point Cloud Library. In each iteration we start our algorithm with the current depth image, which is filtered with statistical outlier removal to reduce noise. The resulting point cloud is segmented into plausible disjoint parts by removing ground points and applying Euclidean clustering. The clusters are passed to the next stage of the pipeline as segmentation result. If no segments are found and the current view does not cover the entire workspace, the robot moves its platform to scan other parts in the scene according to a given scanning strategy. If no segments are found in the scene, the sample collection process is terminated.

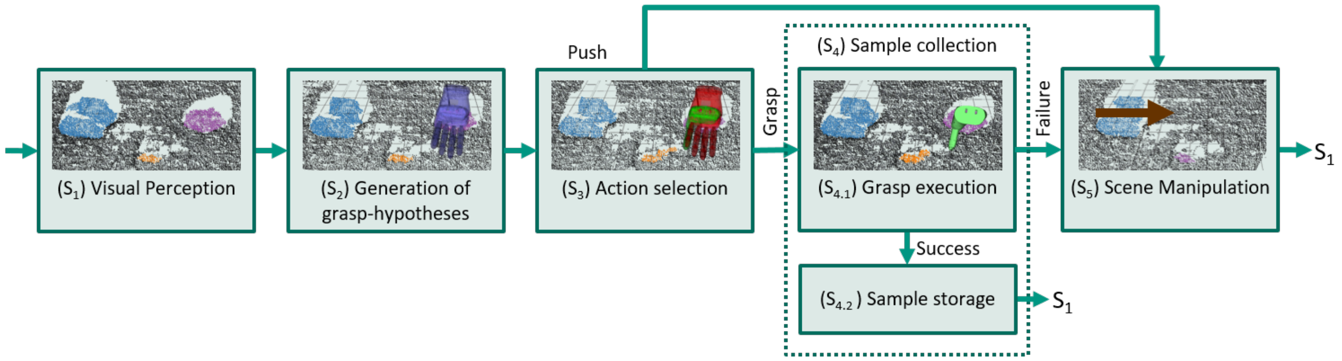
Push
(S$_4$) Sample collection
Grasp
Failure
(S$_1$) Visual Perception
(S$_2$) Generation of grasp-hypotheses
(S$_3$) Action selection
(S$_{4.1}$) Grasp execution
(S$_5$) Scene Manipulation
S$_1$
Success
(S$_{4.2}$) Sample storage
S$_1$

Fig. 2: The five stages of the pipeline are executed in a loop. ($\mathbf{S_1}$) segments the input point cloud, ($\mathbf{S_2}$) generates grasp hypotheses (blue hands), ($\mathbf{S_3}$) filters executable (green) and non-executable (red) hypotheses and decides whether a pushing or grasping action should be performed, ($\mathbf{S_4}$) collects the sample. ($\mathbf{S_5}$) changes the scene through pushing to improve the segmentation.

B. Generation of Grasp Hypotheses

We represent grasp hypothesis as the 6D pose of the tool center point of the hand in the world frame. In addition, we use the fully opened hand as preshape configuration of the hand, since it provides the largest sweep area during execution. This also increases the robustness of grasping actions and helps to deal with inaccuracies in perception and execution.

Grasp hypotheses are generated for each segment of the point cloud. While different approaches can be used to generate these hypotheses, an approach with low computational cost is required to deal with resource constraints mentioned above. To this end, the points of each segment are projected onto the supporting plane and oriented 2D bounding boxes of the projected points are calculated. Each 2D box is extruded by the height of its segment to form a 3D box, one side of which is constrained to the supporting surface. This 3D box is calculated twice: Once for all points in the segment and once after α percent of the point set has been trimmed from the end of each of the axes of the first box. This eliminates outliers and significantly increases the robustness of generated grasp hypotheses. The resulting box is only used to generate grasp hypotheses if its expansion is small enough to allow its grasping by the hand. The 6D pose representing the grasp hypothesis is aligned to the axes parallel to the supporting plane and placed at the center of the box. Due to sensor noise, incorrect segmentation can occur resulting in merging several segments into a single one, splitting a segment or detecting noise as a segment. Such artifacts occur regularly and lead to incorrect or missing grasping hypotheses. In order to remove these artifacts, we apply spatio-temporal filtering over the grasp hypotheses, which are generated from the last κ time frames. To this end, the grasp hypotheses from the last κ time frames are accumulated into the set θ. For each hypothesis $g \in \theta$, the set θ_g of adjacent hypotheses is determined. This set is formed by all hypotheses $\hat{g} \in \theta$ with a Euclidean distance $d_T(g, \hat{g})$ less than ϵ_T and an angular distance $d_A(g, \hat{g})$[1] less than ϵ_A. Thus, the set of neighboring grasp hypotheses is defined by

$$\theta_g = \{\hat{g} \,|\, \hat{g} \in \theta,\, d_T(g,\hat{g}) < \epsilon_T,\, d_A(g,\hat{g}) < \epsilon_A\} \ .$$

[1]The minimal absolute rotation around any axis required to transform g into $\hat{g}$ in radian.

Next, the filtered set θ' of grasp hypotheses is created by selecting only elements that maximize $|\theta_g|$ within their own cluster, and discarding these local maxima if $|\theta_g| < l_{min}$ to reduce noise and outliers[2]. This step is akin to non-maximum suppression and results in median filtering over the cluster, as shown in Fig. 3. In addition to the noise and outlier reduction, the number of grasp hypotheses and thus the number of actions to be considered for execution is reduced, which leads to a reduction of the computing resources required for action selection.

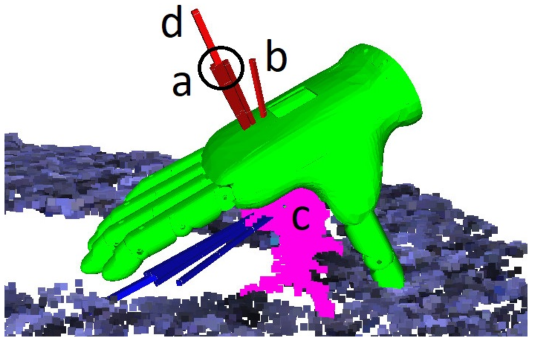

Fig. 3: A cluster of grasp hypotheses (represented by the coordinate frames in circle a) and an outlier hypotheses (b) are generated from the segment (c). The outlier (b) is discarded and (d) is selected to represent cluster (a). The green hand visualizes the pose resulting from (d).

C. Action selection

At this stage of the pipeline a decision must be made to determine the next action to be taken. This means to select whether (1) to execute a grasping action or (2) to execute a pushing action to change the current scene.

To make this decision, we build the set of failed grasps θ_{failed} over several iterations of the pipeline, i. e. grasp hypotheses that did not lead to successfully picking up a stone in previous iterations. Using this set, we construct the set of executable grasp hypotheses θ_{exec} considered for execution by discarding grasping hypotheses from the filtered set θ' in two

[2]$|\cdot|$ denotes the set cardinality

successive steps: First, all hypotheses with a Euclidean distance below $\epsilon_{discard}$ to any element of θ_{failed} are removed. This prevents the repeated execution of grasping actions that have led to failure in previous iterations. Second, grasp hypotheses that are kinematically unreachable for the robot are discarded. The resulting set θ_{exec} only contains grasp hypotheses that are executable by the robot and have not led to failure in previous time steps.

Since the primary goal is to collect the stones, our approach prefers to execute grasping actions rather than changing the scene through pushing actions. Therefore, grasping actions are executed as long as θ_{exec} is not empty.

D. Executing Grasping Actions

Selecting the grasp hypothesis to be executed is done in two steps: First, θ_{exec} is divided into two subsets: θ_{reloc}, which contains all grasp hypotheses that require a change of the position of the robot's mobile base, and $\theta_{noreloc}$, which contains the remaining hypotheses. Since moving the robot's mobile base can lead to inaccuracies, we avoid moving the platform as much as possible. Therefore, hypotheses from θ_{reloc} are only considered in the following step, if $\theta_{noreloc}$ is empty. Among the hypotheses considered, the robot executes the one that maximizes the distance to the joint limits along the entire trajectory, since this ensures a smooth execution.

During execution, the hand is moved above the grasp hypothesis and then downwards until contact is detected with the 6D force/torque sensor mounted at the wrist. After a contact is detected, the robot executes a grasping trajectory that coordinates wrist rotation, hand distance to the contact point, and finger closure. This trajectory performs a sweeping motion of fingers and thumb across the contact surface and compensates for pose uncertainties of the target object.

During grasping and transport to the storage container, the robot continuously checks the force/torque sensor and hand joint angles to determine if an object is in the hand. If the object was not picked up or is dropped during transport, the execution is considered a failure and the corresponding grasp hypothesis is added to θ_{fail}. If no failure is detected, the robot continues with another iteration of the five-step pipeline or stops if enough stones have been collected. In case of failure, θ' is filtered as described in section III-C. If the grasps remain in θ_{exec} this stage is restarted, otherwise a scene manipulation is triggered.

E. Executing Pushing Actions

This stage of the pipeline is triggered if either (1) grasp hypotheses are available ($|\theta'| > 0$), but none is considered for execution ($|\theta_{exec}| = 0$) or (2) there are segments in the scene but they are too large to be grasped. In both cases, pushing actions toward the center region of the workspace are generated. Pushing towards the center prevents stones from being pushed outside of the working area. In the first case, an element of θ' is randomly selected and a pushing action on the corresponding segment is executed in order to make the object graspable by moving and rotating it. In the second case, a segment could represent an under-segmented pile of stones or one large boulder. In order to verify this, pushing actions are used to try to split the segment into smaller parts and thus improving the segmentation and possibly enable grasping. A pushing action is generated by selecting a random segment and trying to push through a random part of the segment.

The execution of the generated pushing action fails, if a high force is detected by the wrist-mounted force/torque sensor. In case of such a failure, the segment is too heavy to be moved by the robot and a new pushing action is generated on the same segment, which only intersects the segment's border. This second pushing action allows to separate loose stones from too heavy or immobile piles of stones or large boulders. If the second pushing action also fails, the segment is marked as too heavy and is avoided in further iterations. If any of the pushing actions is successful, the next iteration of the pipeline is executed.

IV. Evaluation

We present two different evaluations of the proposed approach: (1) A qualitative evaluation of the stone collection process and (2) an evaluation of the computation time required for the approach. A video showing the experiments is attached to this contribution.

In all experiments we choose the point cloud trimming parameter $\alpha = 5\,\%$, collected hypotheses over the last $\kappa = 10$ time frames, treated clusters with less than $l_{min} = 5$ candidates as outliers, considered hypotheses with a translation difference below $\epsilon_T = 25\,\mathrm{mm}$ and a rotation difference below $\epsilon_A = 0.1\,\mathrm{rad}$ as belonging to the same cluster and discarded hypotheses within $\epsilon_{discard} = 50\,\mathrm{mm}$ of any element of θ_{fail}. These parameters depend on the execution accuracy, the size of the hand, noise of the camera and the accuracy of the extrinsic camera calibration. The values were empirically determined based on real experiments and have only have to be determined once for a given hardware setup. The trajectory for grasping depends on the hand and is recorded during a teach-in phase before experiments are performed.

A. System Setup

In our experiments, we use the humanoid robot ARMAR-6, which is equipped with two 8 DOF arms with wrist-mounted 6D force/torque sensors and a holonomic platform (see Fig. 4 and [18]).

The robot is equipped with an underactuated humanoid five-finger hand. Although such a five-finger hand is more complex than a parallel yaw gripper in terms of mechanics and the required control strategy, it allows more contact points with the grasped object, which makes grasping more stable. The underactuated hand facilitates stable grasps by equally distributing the contact forces between all fingers. In pre-experiment tests we measured the force required to push a pile of stones in our test scenario. During these tests we measured forces up to $150\,\mathrm{N}$, which are due to the high friction caused by the rough surface of the stones and gravel. To prevent potential damage to the hand, we mounted a rod which is used for pushing actions to the left arm. A container to collect the

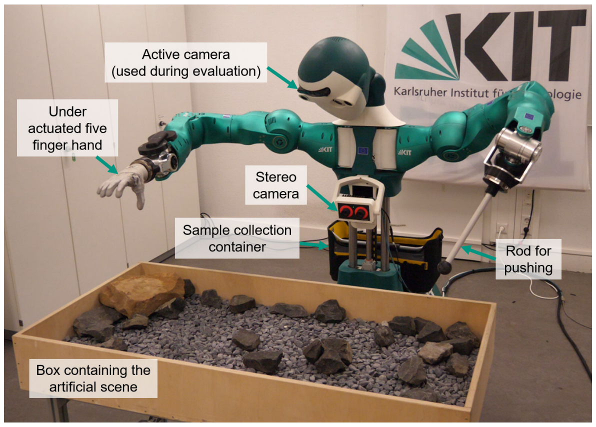
Active camera
(used during evaluation)
Under actuated five finger hand
Stereo camera
Sample collection container
Rod for pushing
Box containing the artificial scene

Fig. 4: ARMAR-6 in front box with stones. The robot has to collect several loose stones as samples and put them into the sample collection container.

grasped stones is placed on the back of the robot. The robot uses a stereo camera system for visual perception.

We conducted two experiments with different computing hardware. In the first experiment, we use a pico-ITX board with an Intel(R) Atom(TM) CPU E3845 @ 1.91GHz and 8GB RAM. In the second experiment we compare the execution time to a computer with an Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz and 31GB RAM. We denote these two computing hardware setups as $PITX$ and PC.

We use a Roboception rc_visard 65 stereo vision system (S). This camera offers three resolutions at different frame rates: (1) S_H: 640×480@3 Hz, (2) S_M: 320×240@15 Hz and (3) S_L: 214×160@25 Hz. As trade-off between frame-rate and quality of the depth image, we use the medium camera resolution S_M for the first experiment. in the second experiment we compare all three resolutions with the active camera of ARMAR-6 (PrimeSense Carmine 1.09 camera, 640×480@30 Hz (A)).

The approach is implemented in C++ using the robot framework ArmarX ([19]). The implementation is mostly sequential. Stage $\mathbf{S_3}$ through $\mathbf{S_5}$ run sequentially and $\mathbf{S_1}$ and $\mathbf{S_2}$ run in parallel them.

B. Sample Collection

This evaluation shows the ability of our approach to collect all stones in an area of interest. For all experiments we use the $PITX$ board and the stereo camera S_M. The setup is shown in Fig. 4.

In the first scenario (see Fig. 5), the robot has the task to pick up all five stones in the region of interest. We place the robot in front of a box containing gravel and stones and allow it to move its mobile base in front of the box. To set up the scene randomly and remove bias, all stones are placed in a small box which is turned upside down to put the stones in a pile. Since some of the stones slip from the stack when the box used for placing is removed, some stones are separated from the stack. We performed the task five times and the robot was always able to pick up all five samples with an average of 11 grasping and 1.6 pushing actions. The coefficient of variance for number grasping actions (42 %, min 6, max 18) and pushing actions (122 %, min 1, max 4) is very high. This shows that the number of required actions depends very much on the specific stone arrangement. For example, a stone rolling from the stack after the box has been removed can drastically decrease the number of required actions. If pushing was prohibited, the robot could pick up an average of 1.4 stones. This result shows that the interactive perception significantly supports the successful grasping of stones. Fig. 5 shows an example for the separation of a pile of stones by pushing.

In the second scenario (see Fig. 6) we evaluated if the robot is able to separate a stone next to a large heavy boulder and collect it. For this purpose, a smaller stone is placed next to a large boulder that is too heavy for the robot to push. Although our IP strategy is primarily aimed at separating piles of stones well, we observed that it generalizes to separating stones from large boulders in a limited way: If the stone is on the side of the boulder closest to the robot, our approach succeeds in reliably separating it. If the boulder occludes the stone or is on the connecting line between the stone and the center of the working area, the separation fails since the generated pushing action always intersects the boulder and thus fails.

C. Computation Cost

We evaluate the time needed to generate grasping hypotheses and plan the next action. We compare the effect of using the active camera A and the three different resolutions offered by the passive stereo camera S on the computing time when using the computer PC and the time required when generating hypotheses using S_M and the $PITX$ board. Table I reports the time needed to generate grasp hypotheses as well as the time for the two sub-steps, the removal of outliers and the generating and filtering grasp hypotheses. In all five cases, the same scene consisting of ten graspable objects is used and the processing time is averaged over 30 s. The comparison of PC-A with PC-S_H, i.e. the PrimeSense camera of the ARMAR-6 with the Roboception stereo system, shows that the time for removing outliers increases when switching from an active to a passive camera system. This is due to the fact that the stereo camera system has a higher level of noise. A comparison of PC-S_H, PC-S_M and PC-S_L shows that a reduction in resolution also reduces the time required to remove outliers, while the time required to filter the hypotheses increases slightly. A lower resolution leads to more noise and results in more grasp hypotheses, which have to be processed in the following filter step. This increases the computation time when switching from PC-S_M to PC-S_L. When switching from PC to $PITX$ for the camera setup S_M the lower computation resources increase the average time to generate a grasp hypothesis. The resulting time of 8.8 ms to generate one grasp hypothesis is sufficient for real time execution, since the system does not have to wait until new grasp hypotheses are available.

During execution on $PITX$, our approach requires on average 1.6 s for action selection, 3.5 s for selecting the grasp hypothesis to execute and 1.6 s for generating a pushing action, resulting in an average of 4.2 s for planning and selecting the next action. This action selection includes filtering out

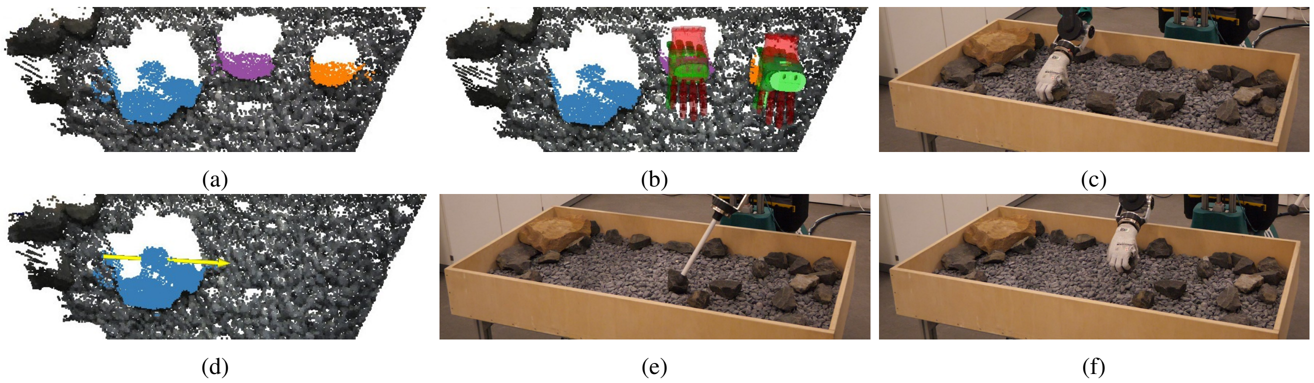

Fig. 5: The robot collects already segmented stones, and uses IP to separate and collect the remaining stones. The approach (a) first segments the scene, (b) generates grasp hypotheses (green: executable, red non-executable) for already separated stones (purple, orange) and (c) collects them. For the remaining segment (d) a pushing actions (yellow arrow) is generated and (e) executed to separate the pile of stones and enable the robot to (f) collect those stones as well.

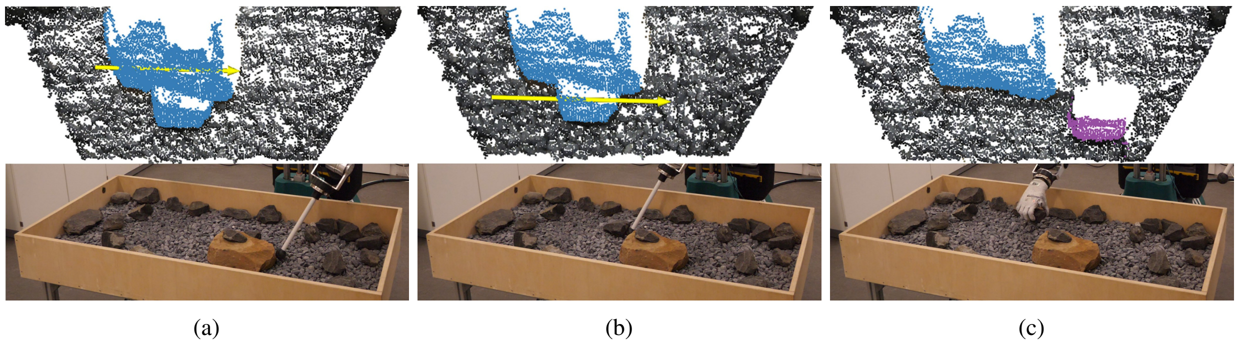

Fig. 6: The robot separates a small stone from a large heavy boulder and picks it up. (a) The first generated pushing action (yellow arrow) fails because the boulder is too heavy. (b) The second pushing action only intersects the border of the large segment and manages to separate the small stone (purple segment). (c) Now the robot is able to pick up the small stone.

Hardware Setup		Average time per grasp (ms)		
Computer	Camera	Total	Outlier removal	Generate and Filter
PC	A	3.00	1.72	1.28
PC	S_H	3.49	2.05	1.45
PC	S_M	2.25	0.66	1.59
PC	S_L	2.50	0.60	1.90
$PITX$	S_M	8.81	3.03	5.78

TABLE I: Computation time per grasp hypothesis for different hardware setups averaged over 30 s. All setups use the same scene as input. Using a stereo instead of an active depth camera increases the calculation time because the point cloud is more noisy. Decreasing the resolution also decreases the calculation time.

grasp hypotheses similar to any from the set θ_{failed}, solving the robot's inverse kinematics to decide if a hypothesis is reachable, determining where to place the robots platform and generating pushing actions.

Considering that action execution in average takes 44.1 s, we consider this well within acceptable margins.

V. Conclusion

We have presented a robust approach that integrates visual segmentation, generating grasp hypotheses, interactive perception, and the execution of grasping and pushing actions. Our approach requires only a low amount of computing resources and can therefore be used in setups where computation resources are is limited (e.g. to an Intel(R) Atom(TM) CPU E3845 @ 1.91GHz) such as mobile robot systems for space exploration. The evaluation shows the effectiveness of the interactive perception strategy, since the robot was always able to collect all five stones within the working area, compared to an average of 1.4 stones if pushing was prohibited. This shows that solving of such challenging problems under severe resource constraints is possible by integrating information gained from physical interaction with the environment and validates the IP approach, showing that scene interaction can improve task success rates.

Future work will focus on optimizing the action selection process, developing a more sophisticated segmentation strategy capable of tracking target objects and transfer and apply the presented methods to the Light Weight Rover Unit [20] in an outdoor sample collection scenario in the context of planetary exploration. In addition, it is interesting to investigate a machine learning-based method that uses energy-efficient hardware tailored to neural networks.

References

[1] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2014.

[2] S. Caldera, A. Rassau, and D. Chai, "Review of deep learning methods in robotic grasp detection," *Multimodal Technologies and Interaction*, vol. 2, no. 3, p. 57, 2018.

[3] P. Song, Z. Fu, and L. Liu, "Grasp planning via hand-object geometric fitting," *The Visual Computer*, vol. 34, no. 2, pp. 257–270, 2018.

[4] N. Vahrenkamp, E. Koch, M. Wachter, and T. Asfour, "Planning high-quality grasps using mean curvature object skeletons," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 911–918, 2018.

[5] Y. Yu, Z. Cao, S. Liang, Z. Liu, J. Yu, and X. Chen, "A grasping CNN with image segmentation for mobile manipulating robot," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2019, pp. 1688–1692.

[6] P. Schmidt, N. Vahrenkamp, M. Wächter, and T. Asfour, "Grasping of unknown objects using deep convolutional neural networks based on depth images," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6831–6838.

[7] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 769–776.

[8] X. Zhou, X. Lan, H. Zhang, Z. Tian, Y. Zhang, and N. Zheng, "Fully convolutional grasp detection network with oriented anchor box," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7223–7230.

[9] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. Aparicio, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.

[10] 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.

[11] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme, "Interactive perception: Leveraging action in perception and perception in action," *IEEE Transactions on Robotics*, pp. 1–19, 2017.

[12] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4238–4245.

[13] L. Berscheid, P. Meiner, and T. Krger, "Robot learning of shifting objects for grasping in cluttered environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 612–618.

[14] M. Danielczuk, A. Kurenkov, A. Balakrishna, M. Matl, D. Wang, R. Martín-Martín, A. Garg, S. Savarese, and K. Goldberg, "Mechanical search: Multi-step retrieval of a target object occluded by clutter," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1614–1621.

[15] H. Zhang, X. Lan, S. Bai, L. Wan, C. Yang, and N. Zheng, "A multi-task convolutional neural network for autonomous robotic grasping in object stacking scenes," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6435–6442.

[16] C. Bersch, D. Pangercic, S. Osentoski, K. Hausman, Z.-C. Marton, R. Ueda, K. Okada, and M. Beetz, "Segmentation of textured and textureless objects through interactive perception," in *RSS Workshop on Robots in Clutter: Manipulation, Perception and Navigation in Human Environments*, 2012.

[17] D. Schiebener, A. Ude, and T. Asfour, "Physical interaction for segmentation of unknown textured and non-textured rigid objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 4959–4966.

[18] T. Asfour, R. Dillmann, N. Vahrenkamp, M. Do, M. Wächter, C. Mandery, P. Kaiser, M. Kröhnert, and M. Grotz, "The karlsruhe armar humanoid robot family," in *Humanoid Robotics: A Reference*, A. Goswami and P. Vadakkepat, Eds. Springer Netherlands, 2019, pp. 337–368.

[19] N. Vahrenkamp, M. W"achter, M. Kr"ohnert, K. Welke, and T. Asfour, "The Robot Software Framework ArmarX," *Information Technology*, vol. 57, no. 2, pp. 99–111, 2015.

[20] P. Lehner, S. Brunner, A. Dömel, H. Gmeiner, S. Riedel, B. Vodermayer, and A. Wedler, "Mobile manipulation for planetary exploration," in *IEEE Aerospace Conference*, 2018, pp. 1–11.