IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), Abu Dhabi, UAE, 2020.

Autonomous Wall Building with a UGV-UAV Team at MBZIRC 2020

Christian Lenz*, Max Schwarz*, Andre Rochow, Jan Razlaw,
Arul Selvam Periyasamy, Michael Schreiber, and Sven Behnke

***Abstract*— Constructing large structures with robots is a challenging task with many potential applications that requires mobile manipulation capabilities. We present two systems for autonomous wall building that we developed for the Mohamed Bin Zayed International Robotics Challenge 2020. Both systems autonomously perceive their environment, find bricks, and build a predefined wall structure. While the UGV uses a 3D LiDAR-based perception system which measures brick poses with high precision, the UAV employs a real-time camera-based system for visual servoing. We report results and insights from our successful participation at the MBZIRC 2020 Finals, additional lab experiments, and discuss the lessons learned from the competition.**

I. Introduction

Mobile manipulation is needed to handle objects in large work spaces, e.g. for constructing structures. While ground-based mobile manipulation has received considerable research attention, robotic aerial manipulation is still in its infancy. The Mohamed Bin Zayed International Robotics Challenge (MBZIRC) 2020[1] posed tasks for robot teams in a demanding outdoor setting. In its Challenge 2, participants were required to build walls out of supplied bricks, both with a UGV and a team of up to three UAVs. The task setting was particularly interesting, because it required complete autonomy, robustness under real-world outdoor conditions with harsh sunlight and wind, and independence from any outside reference system besides the globally available GPS.

In this work, we describe our entry to the MBZIRC 2020 Finals, which consists of a UGV-UAV team (see Fig. 1). In addition to describing our integrated systems for solving the tasks set by the competition and discussing lessons learned, our technical contributions include:

- a flexible and precise magnetic gripper system for large objects addressing the unique design constraints on UAVs,
- a robust and efficient vision-based detection and pose estimation module for box-shaped objects,
- a laser-based pose estimation and registration module for the UGV, and
- a highly space- and time-efficient box storage system for UGVs.

*: equal contribution.

This work has been supported by a grant of the Mohamed Bin Zayed International Robotics Challenge (MBZIRC).

Institute for Computer Science VI, Autonomous Intelligent Systems, University of Bonn, Endenicher Allee 19a, 53115 Bonn, Germany, {lenz, schwarz}@ais.uni-bonn.de

[1] http://mbzirc.com/

Fig. 1. Our UAV Lofty (left) and UGV Bob (right) during the MBZIRC 2020 Finals.

II. MBZIRC 2020

In Challenge 2 of the MBZRIC 2020 competition, a team of one UGV and up to three UAVs had to pick, transport, and place bricks to build a wall. Four different brick types with 20×20 cm cross-section were used: Red (30 cm length, 1 kg), green (60 cm, 1.5 kg), blue (120 cm, 1.5 kg), and orange (180 cm, 2 kg). Each type of robot had a designated pick-up and place area inside the arena (40×50 m). Fig. 2 shows the arrangement of the bricks for the UGV and UAVs at the beginning of the task. Both robots had to build the first wall segment using only orange bricks. For the remaining segments (one for the UGV and three for the UAVs), a random blueprint defining the order of the red, green, and blue bricks was provided some minutes before the competition. Points were granted for correctly placed bricks. The UGV could archive between 1 to 4 points per brick (45 bricks in total); the bricks placed by an UAV counted between 3 to 16 points (46 bricks). The time limit for this challenge was 25 min. All tasks had to be performed autonomously to archive the perfect score. The teams were allowed to call a reset at any time to bring the robots back to the starting location. Resets did not result in a point penalty, but no extra time was granted.

III. Related Work

UGVs for Wall Building: The application of robots for wall-building has a long history [1]. One particularly impressive example is the work of Dorfler et al. [2], who developed a heavy mobile bricklaying robot for the creation of free-form curved walls. An alternative for creating free-form walls is on-site 3D printing with a large manipulator arm [3].

UGVs for Disaster Response: Our work mostly relates to disaster response robotics, where protective or otherwise functional structures have to be built quickly and with minimal human intervention. The DARPA Robotics Challenge [4] established a baseline for flexible disaster response robots.

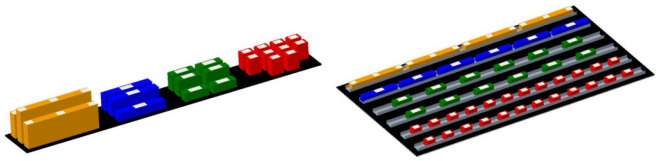

Fig. 2. Brick pickup arrangement for the UGV (left) and UAVs (right).

In comparison to these and to more recent disaster-response robots such as Centauro [5], our system has a much higher degree of autonomy, but is much more specialized for the task at hand.

Aerial Manipulation: In recent years, aerial manipulation has become a research focus [6]. Complex systems with fully actuated multi-DoF robotic arms have been built [7], [8]. Lindsey et al. [9] demonstrated the assembly of structures with teams of small UAVs. This work relied on an external motion capture system and self-locking magnetic part connectors. Goessens et al. [10] present a feasibility study of constructing real-scale structures with UAVs, which is based on self-aligning Lego-like brick shapes.

A predecessor of our work is Challenge 3 of the last MBZIRC edition in 2017, where a team of UAVs was supposed to collect discs. Our entry [11] was quite successful and reached a third place in this challenge, behind ETH Zurich [12] and CTU Prague, UPENN and UoL [13]. In comparison, the 2020 edition of MBZIRC featured much heavier and larger objects, which could only be grasped on a specific spot and had to be placed in a specified pose. To this end, we designed a magnetic gripper that is guided using visual servoing and has five passive DoFs that allow flexibility during grasping but facilitate rigid and precise placement.

IV. UGV Solution

We build our ground robot Bob based on our very successful UGV which won the first MBZIRC competition [14]. We improved the basis and adapted the manipulator and sensors for the new challenge. Since 45 bricks had to be picked, transported, and placed in 25 min to obtain a perfect score, we developed our UGV to store as many bricks as possible. We decided to use a 3D LiDAR as the main sensor to detect and localize the piles of bricks and partially built wall.

A. Hardware Design

UGV components include a four-wheeled omnidirectional base, a 6-DoF manipulator arm with custom-made magnetic gripper, a wrist sensor consisting of a 3D LiDAR as well as an RGB camera, and a 3-DoF storage system.

The base has a footprint of 1.9×1.4 m to provide enough space for our storage system. It rolls on direct-drive brushless DC hub motors, controlled by two ODrive driver boards. Since the motors were originally intended for hover boards, i.e. personal conveyance devices, they have enough torque to accelerate the approx. 90 kg UGV. To achieve omnidirectional movement, we coupled each wheel with a Dynamixel H54-200-S500-R servo which rotates it around the vertical axis. The developed base supports driving speeds of up to 4 m/s and precise positioning for manipulation tasks.

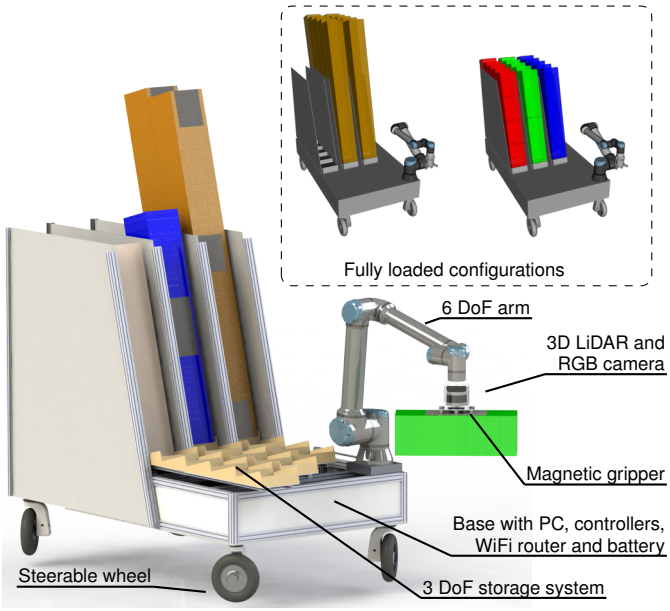

Fig. 3. UGV hardware design. Top right: The storage system is capable of holding either 10 orange bricks (left) or all remaining bricks (20 red, 10 green, 5 blue) (right).

This year's competition required to manipulate objects of up to 1.80 m length and to stack them to a total height of 1 m. Instead of the UR5 mounted on our 2017 robot, we used a UR10e 6-DoF robotic arm, which gives us the benefit of a larger workspace (1.30 m) and enough payload capability (10 kg) to carry the gripper, including the sensors, and the bricks (up to 2 kg). We adapted the arm controller to work with UGV battery power.

Bob's gripper is equipped with eight electromagnet and a contact switch. Since the ferromagnetic parts of the bricks are very thin (approx. 0.6 mm), we decided to use a larger number of smaller magnets to distribute the contact surface as much as possible while keeping the total gripper size minimal. The switch detects if a brick is securely grasped.

For perceiving the bricks, we mounted a Velodyne VLP-16 3D LiDAR and a Logitech Brio camera on the wrist. The LiDAR is our main sensor for detecting the bricks and for estimating their poses relative to the robot (see Section IV-C). The RGB images are used to detect the wall marker (see Section IV-C.2). A second Logitech Brio camera mounted at the top of the robot provides the operator situation awareness.

We designed a storage system which has three individually actuated storage compartments. Each compartment has five bins to store bricks. The ground plate of each bin is 20,5 cm wide, 20 cm long and is mounted inclined 15° backwards. This inclination forces the bricks to slide in a known pose inside the storage system even if the bricks are grasped imprecise. Hence, we do not need an additional perception system to perceive the current pose of the stored bricks. Side walls hold the bricks in place during UGV movements. The walls are 110 cm high which is sufficient to hold the largest bricks (180 cm long) in place. Furthermore, this system allows to stack multiple small bricks (up to 4 red bricks, and up to 2 green bricks) to increase the number of bricks to be

stored in the system. Overall, the system is capable to store either all large bricks (10 orange), or all remaining bricks (20 red, 10 green, 5 blue, see Fig. 3). Since the storage system exceeds the workspace of the UR10e, each compartment can be moved horizontally (using a Dynamixel Pro L42-10-S300-R and a linear belt drive) to put the desired bin in reach of the arm.

The UGV is equipped with a standard ATX mainboard with a quad-core Intel Core i7-6700 CPU and 64 GB RAM. The whole system is powered by an eight-cell LiPo battery with 20 Ah and 29.6 V nominal voltage. This allows the robot to operate for roughly one hour, depending on the task.

Due to resource conflicts when building all robots needed for the MBZIRC 2020 competition, hardware and software component development and testing was initially executed on the modified Mario robot [14]. The larger Bob chassis was assembled on site in Abu Dhabi for the first time.

B. High-level Control

We implemented a high-level controller consisting of a finite-state machine (FSM) generating the robot actions, a database to keep track of every brick relevant for the UGV, and an algorithm computing the time-optimal strategy for a given build order.

The FSM includes 32 different states for locomotion, manipulation, perception, storage logistics, and fallback mechanisms. After executing an action, the resulting database and FSM state was stored to enable quick recovery after a reset.

Since the UR10 arm is very precise, we can manipulate multiple bricks from a stationary position after perceiving the environment just once. Our overall strategy was to minimize locomotion between different positions in front of the piles and the wall. Thus, the plan was to pick up all orange bricks and bring them to the wall at once. After successfully building the orange wall segment, the UGV was to collect all remaining bricks to build the second wall segment. Whereas the orange wall segment (two stacks of 5 bricks each) can be built from two predefined positions, the build order of the second wall segment highly depends on the supplied blueprint and can be optimized to minimize the number of locomotion actions.

We implemented a backtracking algorithm to find the optimum build order. To make this approach feasible regarding runtime, we only consider building the wall from left to right, but allow starting the next layer before finishing the first. Let the longer wall axis (from left to right) be denoted as the x-axis. First, we compute the set of possible place positions by $P = \{x_i + t_x | x_i = \text{center of brick } i\}$. The place pose is shifted by the arm reach $t_x = 0.675\,\text{m}$ to place the robot such that the number of brick placement poses in reach is maximized. Due to the wall structure, we have $7 \leq |P| \leq 35$. We now enumerate all possible ordered sequences $S \subseteq P$. For each $p_i \in S$, we build all bricks which meet the following criteria:

1) The brick was not built already,
2) the brick is in reach based on the position p_i,
3) the brick is fully supported by the ground or previously built bricks, and
4) the left adjacent brick was built.

$S = (p_1, p_2, \dots)$ is a valid solution if all bricks are built. We search for the optimal solution with $|S|$ and d_S minimal, where $d_S = \sum_{i=2}^{|S|} |p_i - p_{i-1}|$, i.e. the shortest path to traverse between all building positions. Pruning sub-trees is used to accelerate the algorithm.

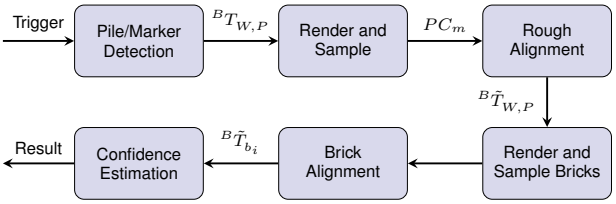

Fig. 4. LiDAR-based brick perception pipeline.

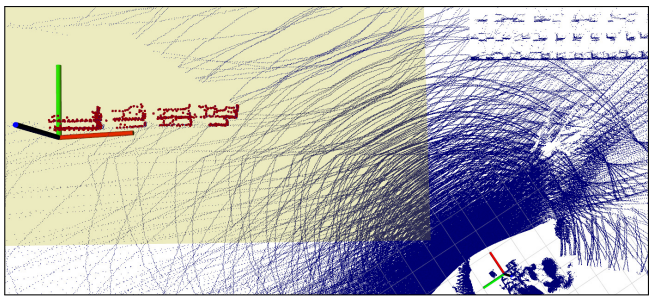

Fig. 5. Top down view for pile detection from LiDAR points (blue). The robot is located at the small coordinate system (bottom). The search area can be restricted using geofencing (yellow rectangle). Detected points are visualized in red and the estimated pile pose is shown.

C. Brick and Wall Perception

When the robot is close to either the pick-up location (called *pile*) or the place location (called *wall*), it needs to localize against these objects and to perform pose estimation of the individual bricks in order to pick them or place new bricks next to them.

Our perception pipeline assumes knowledge of the current state of the world, including a rough idea of the brick poses relative to the pile or wall. The perception pipeline receives this information from the high-level control module.

Depending on the target (pile/wall), the perception pipeline receives an initial guess of the target pose ${}^{B}T_{P}$ or ${}^{B}T_{W}$ w.r.t. the robot's base (B). It also receives the brick pose ${}^{W,P}T_{b_i}$ and brick type $t_i \in \{\text{r, g, b, o}\}$ for each brick i. For initial alignment purposes, the individual brick alignment can be switched off. Finally, bricks can be excluded from the optimization, for example if they are far away and not of interest for the next action.

Figure 4 shows the overall perception process. In both cases, it starts with a rough detection of the target location from further away.

1) Rough Pile Detection: In case of the pile, we know the approximate shape beforehand. We make use of this and search for corresponding measurements using the 3D LiDAR sensor. While doing so, one needs to take care to

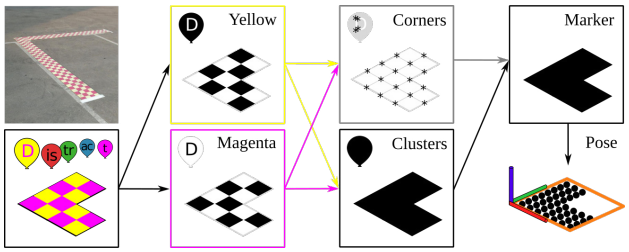

Fig. 6. Wall marker detection. Starting from the input image (Col. 1), two color masks are generated (Col. 2). These masks are used for extracting corners (Col. 3 top) and clustering (Col. 3 bottom). Corners vote for clusters to detect the wall marker (Col. 4 top). Marker pose is estimated using oriented bounding box (in orange around projected points col. 4 bottom).

disambiguate the UGV and UAV piles and other distractors in the arena. The first step in detecting the pile is to confine the search space to a user-defined search area, so-called geofencing. We start by filtering out the points that lie outside of the search area and fit a plane to the remaining points. Next, we filter out the points that lie on the plane or are very close to the plane. The remaining points, shown in Fig. 5, may belong to the pile. After clustering the points and filtering clusters which do not fit the expected pile size, we perform PCA on the remaining cluster to estimate the largest principal component and define the pile coordinate system such that the X axis is aligned with the 2D-projected principal axis and the Z axis points vertically upwards.

2) Marker Detection: After picking up bricks, the next task is finding and estimating the pose of the L-shaped marker indicating where to build the wall (see Fig. 6). Our idea for detecting the marker relies on its distinctive color, pattern and shape best visible in camera images. We start by specifying volumes within the HSV color space corresponding to the yellow and magenta tones of the marker. Now, we exploit the characteristic color composition of the marker to filter out distractors in the image. For that, we generate a color mask using all yellow pixels that are close to magenta pixels and another one for magenta pixels in the vicinity of yellow pixels (Fig. 6 Col. 2). The resulting masks preserve the pattern of the marker which we utilize to filter out further distractors. First, we extract the corners from each mask separately and then search for corners present in close vicinity in both masks. Additionally, we fuse both masks and extract clusters of all masking pixels (Fig. 6 Col. 3). Next, we let each resulting corner vote for its corresponding cluster. The cluster gathering most votes is assumed to be corresponding to the wall marker (Fig. 6 Col. 4 top).

We project each cluster pixel onto the ground plane of the arena and accumulate the resulting point clouds of the previous 10 seconds, since the camera has limited FoV and we can make use of the robot and arm movements to cover more space. After Euclidean clustering, we compute the smallest oriented 2D rectangle around the biggest cluster. The intersection point of the L shape can be found by looking for the opposite corner, which should have the highest distance from all cluster points (see Fig. 6 Col. 4 bottom). Finally, the detection is validated by verifying the measured side lengths.

3) Rendering and Sampling: The next module in the brick perception pipeline (Fig. 4) converts our parametrized world model into 3D point clouds that are suitable for point-to-point registration with the measurements PC_s of the Velodyne 3D LiDAR, which is moved to capture a dense 3D scan of the pile or brick scene. We render the parametrized world model using an OpenGL-based renderer [15] and obtain the point cloud PC_m. Both point clouds are represented in the base-link B. Since we render at a high resolution of 2800×2800 pixels, we downsample the resulting point cloud to uniform density using a voxel grid filter with resolution d = 0.02 m.

4) Rough Alignment: We will now obtain a better estimate ${}^B\tilde{T}_W$ or ${}^B\tilde{T}_P$ of the pile/wall pose. We first preprocess PC_s as follows:

1) Extract a cubic region around ${}^BT_W$ / ${}^BT_P$,
2) downsample to uniform density of using a voxel grid filter with resolution 0.02 m,
3) find and remove the ground plane using RANSAC, and
4) estimate point normals (flipped s.t. they point towards the scanner) from local neighborhoods for later usage.

We then perform Iterative Closest Point (ICP) with a point-to-plane cost function [16] with high correspondence distance, which usually results in a good rough alignment, followed by a point-to-point alignment with smaller correspondence distance for close alignment.

In case the wall marker was detected, we add another cost term

$$E_{dir}({}^B\tilde{T}_W) = (1 - ({}^B\tilde{R}_W \cdot (1\,0\,0)^T)^T \vec{l})^2 \qquad (1)$$

with $\vec{l}$ being the front-line direction and ${}^B\tilde{R}_W$ the rotation component of ${}^B\tilde{T}_W$. This cost term ensures the optimized wall coordinate system is aligned with the marker direction.

The above-defined cost function is optimized using the Ceres solver until either the translation and rotation changes or the cost value change are below termination thresholds ($\lambda_T = 5 \times 10^{-8}$, $\lambda_C = 1 \times 10^{-6}$).

5) Individual Brick Pose Optimization: When the robot is close enough, we can determine individual brick poses. We constrain the following optimization to translation and yaw angle (around the vertical Z axis), since pitch and roll rotations can only happen due to malfunctions such as dropping bricks accidentally. In these cases, the brick will most likely not be graspable using our gripper, so we can ignore these cases and filter them later.

For correspondence information, we re-render the scene using the pose ${}^B\tilde{T}_{W,P}$ obtained from rough alignment. Here, we include the ground plane in the rendering, since we can use it to constrain the lowest layer of bricks. We separate the resulting point cloud into individual brick clouds PC_{bj}.

We now minimize the objective

$$E_{\text{multi}} = \sum_{j=1}^{N} \sum_{i=1}^{M(j)} \frac{1}{M(j)} \|(R(\theta_j)p_{j,i} + t_j - q_{j,i})^T n_{q_{j,i}}\|^2, \qquad (2)$$

where the optimized parameters θ_i and t_i describe the yaw angle and translation of brick i, N is the number of bricks, $M(j)$ is the number of found point-to-point correspondences for brick j, $p_{j,i} \in PC_{bj}$ & $q_{j,i} \in PC_s$ are corresponding

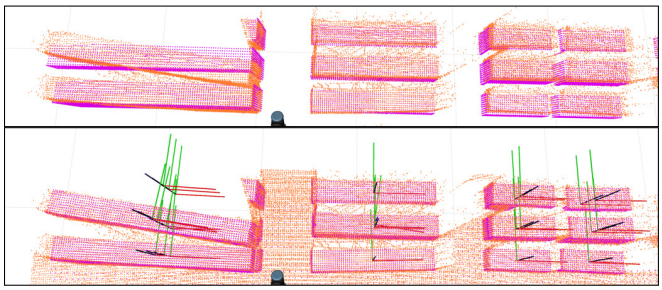

Fig. 7. Precise alignment of individual bricks. Laser measurements are colored orange, model points are shown in purple. Top: Initial solution found by the rough ICP stage. Bottom: Resulting brick poses.

points, and n_q is the normal in point q. This is a point-to-plane ICP objective with separate correspondences for each brick. Correspondences are filtered using thresholds λ_{dot} and λ_{dist} for normal dot products and maximum point distances.

To keep the wall structure intact during optimization, we add additional cost terms for relationships between bricks that touch each other, which punish deviations from their relative poses in the initialization:

$$E^R_{i,j} = \lambda_r \| R(\theta_i)\, {}^B R_{b_i} (R(\theta_j)\, {}^B R_{b_j})^{-1}\, {}^B R_{b_j}\, {}^{b_i} R_B - I \|^2_F, \quad (3)$$

$$E^T_{i,j} = \lambda_t \| t(T(\theta_i, t_i)\, {}^B T_{b_i} (T(\theta_j, t_j)\, {}^B T_{b_j})^{-1}\, {}^B T_{b_j}\, {}^{b_i} T_B) \|^2_2, \quad (4)$$

where $|| \cdot ||_F$ denotes the matrix norm, and λ_r, λ_t are balancing factors. Note that these pairwise cost terms have equal strength for all involved brick pairs.

As in the rough alignment phase, the parameters are optimized using Ceres using the same termination criteria up to a maximum of 20 iterations. The optimization takes around 0.15 s on the onboard computer for one iteration with 20 bricks. In addition, we compute a confidence parameter for each brick as the ratio of found correspondences to expected visible points according to the rendered model. Figure 7 shows an exemplary result of the entire pipeline.

D. Experiments

During the MBZIRC 2020 Finals, our UGV Bob performed in six arena runs. We used the three rehearsal days to get familiar with the arena conditions, fixed Wi-Fi issues, picked bricks in a semi-autonomous way and fine-tuned our perception pipeline. Unfortunately, in the first Challenge 2 competition run we had issues regarding the gripper. We attempted over 15 times picking up an orange brick with very promising perception results but were not successful. We were unable to fix this problem since hardware changes were not allowed during the competition run. In the second competition run we were able to pick and store a green brick successfully, but again scored zero points since our UGV was not able to drive accurately on the slope inside the arena to reach the wall position. Due to limited test time we did not discover this problem earlier. Nevertheless, the points collected by our UAV were enough to secure an overall second place. In the final Grand Challenge, our UGV was assigned to first solve Challenge 3 (fire fighting) to maximize the overall points of our team. After successfully solving Challenge 3, only two minutes were left, which was not enough time to score any points in Challenge 2.

TABLE I

BUILD ORDER OPTIMIZATION

Method	$\|B\|$		d_B [m]		Runtime [s]	
	mean	stddev	mean	stddev	mean	stddev
Optimal	5.0	0.91	3.36	0.97	7.5	29.0
Greedy	5.5	1.10	5.63	1.86	0.0	0.0

Computing the build order (B) using our optimization versus a greedy approach over 1000 randomly generated blueprints. The path length to reach all build positions is denoted as d_B.

After the competition, we evaluated two sub-systems of our UGV in our lab environment. We compared our algorithm for optimizing the build order with a greedy strategy. Using the greedy strategy, we take the best local solution, i.e. given a set of already built bricks, we chose the next build position such that the number of bricks the UGV is able to build is maximal. Table I shows the results of both approaches on 1000 randomly generated blueprints. The optimization reduces the different build positions needed from 5.5 to 5.0 on average and gives an even larger improvement regarding the distance needed to be driven by 2.3 m on average. Performing the optimization takes on average 7.5 s, which is feasible in our use case since it is done just once before the competition run. Nevertheless, it is—as expected—much slower than the greedy approach due to the exponential complexity.

In a second lab experiment, we evaluated the precision and repeatability of picking up bricks from a pile (see Fig. 8). We placed four bricks in front of the robot similar to the competition setup. Each test consists of scanning the piles, picking the brick with the highest confidence, and placing it at a predefined pose. We calculated the mean translation and rotation error compared to a perfectly aligned center-grasped brick. Only a rough estimation of the pile location was provided to the system. We repeated the test ten times while changing the brick horizontal positions by up to 5 cm and the rotation by up to 10° around the vertical axis. Table II shows the mean results per brick. The resulting mean error could be further decreased by investing more time calibrating the whole system; nevertheless, it is sufficient to place the bricks reliably into the storage system. The very low standard deviation in both rotation and translation shows that our perception and grasping pipeline has a high repeatability and is very robust.

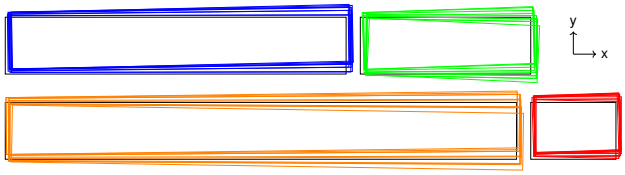

Fig. 8. UGV picking robustness. Ground truth place pose (black) and ten test results for each brick type.

TABLE II
END-TO-END BRICK MANIPULATION PRECISION

Brick	Translation x [cm]		Translation y [cm]		Yaw [°]	
	mean	stddev	mean	stddev	mean	stddev
Red	1.52	0.36	1.49	0.37	1.26	0.80
Green	1.94	0.33	0.67	0.43	1.26	0.68
Blue	1.82	0.49	0.53	0.21	0.76	0.40
Orange	1.34	0.65	0.36	0.39	0.45	0.24

Placement error from perceiving, picking, and placing. Ten tests per color.

V. UAV SOLUTION

Since the initial rules specified a shared wall where UAVs and UGVs could collaborate, we concentrated our efforts on the UGV design. In a late rule revision UGV and UAV walls were separated, making it clear to us that UAV points had to be scored in order to win. Our UAV design thus focused on a minimal solution that could achieve almost full points: We decided to ignore the orange bricks of 1.8 m length, which were intended to be carried by two UAVs. Our system should support the red (0.3 m), green (0.6 m), and blue (1.2 m) bricks.

A. Hardware Design

Because of the weight of the larger bricks and their size, we decided to use a large UAV, the DJI Matrice 600 (M600), for this task. The M600 offers sufficient payload and battery life (roughly 20 min in our configuration).

A key component for aerial manipulation is the robotic gripper. UAVs pose unique constraints when compared with ground-based manipulation. The gripper has to be light-weight in order to fit inside the payload constraints. Furthermore, a certain flexibility and mechanical compliance is desired for two reasons: First, this allows a grasp to succeed even if the approach was not fully precise. Secondly, a rigid connection between the UAV and the ground can be very dangerous, since UAVs usually tightly and very dynamically control their attitude in order to hold position. One can easily imagine situations where the UAV has to drastically change attitude in response to wind gusts and of course hindrance by the gripper system should be limited. However, during the placement phase of the pick-and-place operation, we require very precise control of the target object. Here—at least while the target object is still in the air—we want a rigid attachment to the UAV. To resolve these seemingly contradicting goals, we designed the gripper system to be rigid only while load is applied, i.e. the brick is hanging below the UAV.

Our gripper design (see Fig. 9) consists of four carbon fiber telescopic rods, which hold a plate equipped with 8 electromagnets (similar to the UGV gripper) below the UAV. When the rods are fully extended, the gripper plate is in a fixed pose and can only move upwards. The more the gripper plate is pressed upwards (e.g. due to contact with a brick), the more it can move sideways and rotate due to the gained movement range in each rod. The gripper is equipped with a switch to detect successful grasping.

Since the standard foldable landing legs on the M600 would interfere with the gripper, we replaced them with fixed landing legs (see Fig. 9).

B. Brick and Wall Perception

The competition task involves two perception challenges: finding and precisely localizing the bricks and localizing with respect to the target wall. Similarly to the gripper system, the UAV places unique constraints on the perception system. Because the gripper is mounted directly beneath the UAV, any observation of a brick close to the gripper must be done from the side. The necessary off-center mounting of the sensor severely limits the sensor weight. A 3D-LiDAR as used in the UGV is too heavy. We chose the Intel RealSense D435 RGB-D camera as a primary sensor for its light weight and its capability to work in sunlight. To achieve good coverage of the terrain below the UAV and to be able to observe large parts of the wall during the placement process, we mounted three D435 sensors on the UAV (see Figs. 9 and 10). In contrast to the UGV solution, the UAV solution needs to be real-time capable to allow tracking during approach.

1) Brick Detection & Pose Estimation: The gripper is visible in all camera images and would lead to confusion with bricks. For this reason, we mounted an ArUco marker [17] on it. The marker pose can be efficiently estimated in each of the three cameras and is low-pass filtered to obtain a robust estimate of the gripper pose below the UAV. Pixels in the immediate vicinity to the detected gripper are discarded for the following processing steps.

Since the white patches on the bricks are quite distinctive (see Fig. 10), we use them to detect the bricks and estimate their pose. In a first step, we convert the input image (resolution 1280×720) to the HSV color space. To detect high-saturation pixels (the colored bricks) in the neighborhood, we downsample the input image to half resolution and run a box filter with kernel size 290×290 to obtain a local saturation average $\bar{S}$ and local value average $\bar{V}$. A pixel p is classified as *patch*, if $S(p) < \bar{S}(p) - \lambda_S \wedge V(p) > \bar{V}(p) + \lambda_V$, or, in other words, the saturation is less than the local average and the value (brightness) is larger than the local average, by user-specified thresholds. This simple segmentation method is modeled after the ones used for detecting chessboard patterns and leads to highly robust performance (see Fig. 10).

Contours with exactly four corners (after contour simplification) are processed further: We check that each corner has a *patch* pixel on the inside and a high-saturation pixel on

Fig. 9. UAV hardware design. Left: Full assembly. Right: Magnetic gripper with passive compliance. The four telescopic rods are shown in fully extended configuration.

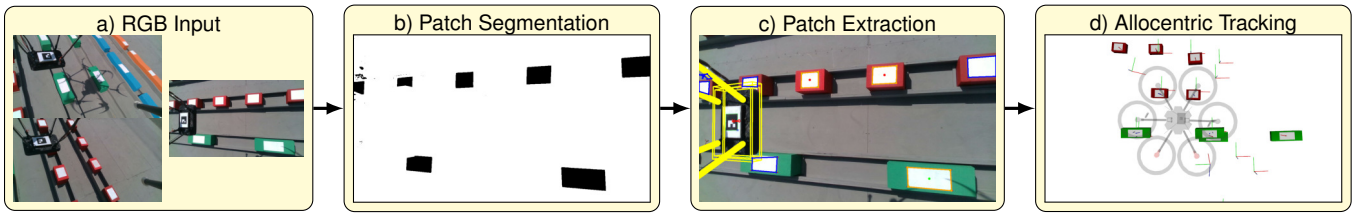

Fig. 10. Camera-based UAV brick perception pipeline. a) Input frames from all cameras. b) White patch segmentation. c) Patch corner extraction & pose estimation. Patch contours in orange (verified) and blue (wrong shape). Brick type is indicated by a colored center point. The gripper is overlaid in yellow. d) Tracking of detections from all three cameras in GPS frame. Detections are shown as bricks, while tracked hypotheses are shown as coordinate axes.

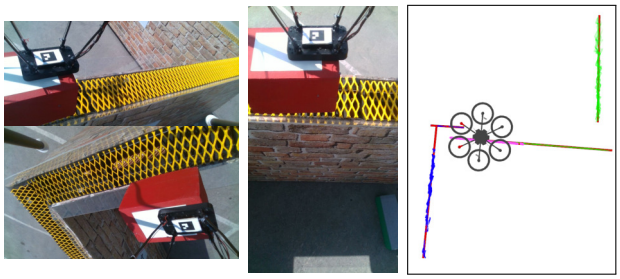

Fig. 11. Wall localization. Left: RGB images from all three cameras. Right: Top-down view of detected wall points per camera (green, blue, purple) and detected wall segments (red lines).

the outside at a specified distance $d = 4$ pixels. The high-saturation pixels on the outside are independently classified into the four possible colors. If all agree, the brick is detected.

Finally, a PnP solver is used to determine the 6D pose of the brick from the recovered 2D-3D correspondences. Here, we assume that the longer side in the 2D image corresponds to the longer brick side in 3D—an assumption which is only violated at extreme viewing angles. To fuse the detections from all three cameras and to track bricks over time, we apply a basic Multi-Hypothesis Tracking (MHT) method with one Kalman filter per hypothesis.

2) Wall Localization: After grasping a brick, the UAV needs to bring it to the wall and place it. Before the competition in Abu Dhabi very little was known about the wall except its geometric shape: Four segments of four meters length and 1.7 m height, arranged in a "W" shape. Especially the top part, which is easily visible from the UAVs perspective, was highly problematic: No information about visual appearance was available, visibility of the top covering (gridding with unspecified mesh size) in our depth sensors was unknown, and later on it would be covered with placed or dropped bricks. We decided to focus on the side walls instead, which where specified as more or less flat surfaces. Especially the side-facing cameras would be able to see the side walls during close approach.

Consequently, our wall perception module estimates the height above ground from the depth image of the downward-facing camera. Points from each camera are then filtered so that only points above 1.0 m and below 1.7 m remain. The data is then projected to 2D, where lines can be extracted using RANSAC. Each line, if fit correctly, corresponds to a side view of one wall segment (see Fig. 11).

The system is initialized with a user-specified initial wall pose, which serves as the search pose. Any time two parallel line segments of valid length with 4 m distance are found, the wall pose is updated. Under the assumption that the wall did not rotate 180°, this is unambiguous.

During close approach, the UAV targets a specific place pose on one of the wall segments. The detected segment closest to the expected segment pose is identified and the goal position is projected onto this segment.

C. High-level Control

Similar to the UGV, the high-level control module is implemented in a FSM framework. It is supplied with the target wall pattern as defined by the organizers of the competition. The basic cycle of events is designed as follows:

1) Fly to the last known pile pose and fly a search pattern until the next brick requested by the pattern is found.
2) Grasp the brick and lift it.
3) Fly to the target position (relative to the last known wall pose) and look for a wall segment.
4) Approach the projected position on the wall segment and place the brick.

Similarly to our MBZIRC 2017 approach [11], we utilize a "cone of descent" during grasping and placement, in which the UAV is allowed to descend towards the target pose. If it drifts outside of the cone, it has to stay at that height until the disturbance has been rejected. The cone angle is 10° with a hysteresis of 3° to prevent oscillations. The cone was shifted such that at the target height it had a radius of 9 cm, which was determined as the maximum deviation that would still allow successful magnetic grasping.

D. Experiments

During the MBZIRC 2020 Finals, our UAV Lofty performed in six arena runs: three rehearsal runs, two Challenge 2 runs, and the final Grand Challenge run.

We used the rehearsal runs to get used to the conditions in Abu Dhabi and continuously improved our pick success rate. During our first Challenge 2 run, we only picked one red and one green brick due to difficulties with our magnetic gripper. Both bricks were dropped close to, but not on the wall due to wall tracking problems. The wall tracking module had not been tested until this point due to short development time and lack of suitable testing opportunities at the competition. After improving our gripper overnight, we managed to pick four red bricks and one green brick and placed two red bricks successfully during our second Challenge 2 run. The other bricks were sadly dropped right next to the wall due to another wall tracking problem. This run was scored as

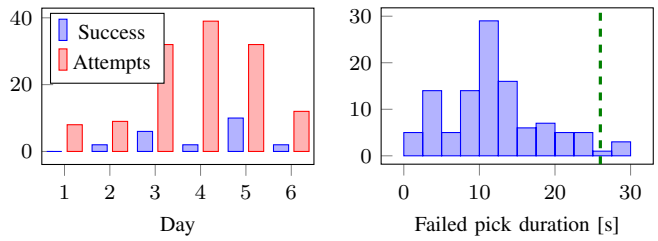

Fig. 12. Picking robustness. Left: Success rate over the duration of the competition. Right: Histogram of failed pick durations. The average successful pick duration is shown in green.

1.33 points, which secured a second place in Challenge 2, next only to the Prague-Pennsylvania team.

In the Grand Challenge, Lofty managed to pick a red brick, but placed it a bit too high and it fell off the wall. After a longer pause to allow our Challenge 1 UAV to operate, it started again and picked up a green brick. Sadly, it falsely detected a W-shaped wall behind the arena netting. Due to a rushed setup sequence, the geofencing was not configured correctly and did not prevent Lofty from flying into the net. After a short unsuccessful rescue attempt during a reset, we had to leave it there for the rest of the Grand Challenge.

Overall, Lofty executed 132 pick attempts in Abu Dhabi, of which 22 were successful, which gives a success rate of 16.7%. Since a failed attempt took 12 s on average, this limited the number of attempts we had for placing bricks on the wall. The number of pick attempts increased over the duration of the competition (see Fig. 12) as the rest of the system became more robust. There are two peaks in the duration histogram for failed picks: One at roughly three seconds which corresponds to tracking failures during the initial approach, and a larger one around 10 s, which corresponds to misaligned picks or magnet failures.

VI. Conclusion

We take the opportunity to identify key strengths and weaknesses of our system and development approach. We also want to identify aspects of the competition that could be improved to increase scientific usefulness in the future.

First of all, this edition of the MBZIRC suffered from low team performance, to the extent that the Grand Challenge price money was not paid out on recommendation of the jury. This underperformance of all teams points to systematic issues with the competition. From the perspective of participants, we think the late changes of the rules have certainly contributed to this situation. A pre-competition event such as the Testbed in the DARPA Robotics Challenge can help to identify key issues with rules and material early in the competition timeline. Another issue was the required effort to participate in all the different sub-challenges. MBZIRC 2020 defined seven different tasks—ideally, one would develop specialized solutions for all of these. Focusing the competition more on general usability, i.e. defining multiple tasks that can and should be completed by one platform, would lower the barrier for participants.

Regarding our system, we saw very little problems with our hardware design—both robots could have scored their theoretical maximum. After solving initial problems with our magnets, especially the passive UAV gripper turned out to be an advantage over other teams, who could not manipulate the heavier bricks. The UGV brick perception provided reliable brick poses during the competition and during lab experiments.

The biggest issue shortly before and during the competition was unavailable testing time. Robust solutions require full-stack testing under competition constraints. Since we postponed many design decisions until the rules were settled, our complex design could not be tested fully. In the end, simpler designs with fewer components, which would have required less thorough testing, could have been more successful in the short available time frame.

We presented a UGV-UAV system for autonomous wall building, which successfully competed at the MBZIRC 2020. We will continue research into aerial and terrestrial manipulation and further UAV-UGV cooperation.

References

[1] A. H. Slocum and B. Schena, "Blockbot: A robot to automate construction of cement block walls," *Robotics and Autonomous Systems*, vol. 4, no. 2, pp. 111–129, 1988.

[2] K. Dörfler, T. Sandy, M. Giftthaler, F. Gramazio, M. Kohler, and J. Buchli, "Mobile robotic brickwork," in *Robotic Fabrication in Architecture, Art and Design*, Springer, 2016, pp. 204–217.

[3] S. J. Keating, J. C. Leland, L. Cai, and N. Oxman, "Toward site-specific and self-sufficient robotic fabrication on architectural scales," *Science Robotics*, vol. 2, no. 5, 2017.

[4] E. Krotkov, D. Hackett, L. Jackel, M. Perschbacher, J. Pippine, J. Strauss, G. Pratt, and C. Orlowski, "The DARPA robotics challenge finals: Results and perspectives," *Jnl. Field Rob.*, vol. 34, no. 2, 2017.

[5] T. Klamt *et al.*, "Flexible disaster response of tomorrow: Final presentation and evaluation of the CENTAURO system," *IEEE Robotics and Automation Magazine*, vol. 26, no. 4, pp. 59–72, 2019.

[6] F. Ruggiero, V. Lippiello, and A. Ollero, "Aerial manipulation: A literature review," *Rob. and Automation Letters*, vol. 3, no. 3, 2018.

[7] F. Huber, K. Kondak, K. Krieger, D. Sommer, M. Schwarzbach, M. Laiacker, I. Kossyk, S. Parusel, S. Haddadin, and A. Albu-Schäffer, "First analysis and experiments in aerial manipulation using fully actuated redundant robot arm," in *IROS*, IEEE, 2013.

[8] S. Kim, S. Choi, and H. J. Kim, "Aerial manipulation using a quadrotor with a two DOF robotic arm," in *IROS*, 2013.

[9] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction with quadrotor teams," *Autonomous Robots*, vol. 33, no. 3, pp. 323–336, 2012.

[10] S. Goessens, C. Mueller, and P. Latteur, "Feasibility study for drone-based masonry construction of real-scale structures," *Automation in Construction*, vol. 94, 2018.

[11] M. Beul, M. Nieuwenhuisen, J. Quenzel, R. A. Rosu, J. Horn, D. Pavlichenko, S. Houben, and S. Behnke, "Team NimbRo at MBZIRC 2017: Fast landing on a moving target and treasure hunting with a team of micro aerial vehicles," *Jnl. Field Rob.*, vol. 36, no. 1, 2019.

[12] R. Bähnemann, M. Pantic, M. Popović, D. Schindler, M. Tranzatto, M. Kamel, M. Grimm, J. Widauer, R. Siegwart, and J. Nieto, "The ETH-MAV team in the MBZ International Robotics Challenge," *Jnl. of Field Robotics*, vol. 36, no. 1, pp. 78–103, 2019.

[13] V. Spurnỳ, T. Báča, M. Saska, R. Pěnička, T. Krajník, J. Thomas, D. Thakur, G. Loianno, and V. Kumar, "Cooperative autonomous search, grasping, and delivering in a treasure hunt scenario by a team of UAVs," *Jnl. Field Rob.*, vol. 36, no. 1, 2019.

[14] M. Schwarz, D. Droeschel, C. Lenz, A. S. Periyasamy, E. Y. Puang, J. Razlaw, D. Rodriguez, S. Schüller, M. Schreiber, and S. Behnke, "Team NimbRo at MBZIRC 2017: Autonomous valve stem turning using a wrench," *Journal of Field Robotics*, vol. 36, no. 1, 2019.

[15] M. Schwarz and S. Behnke, "Stillleben: Realistic scene synthesis for deep learning in robotics," in *ICRA*, 2020.

[16] K.-L. Low, *Linear least-squares optimization for point-to-plane ICP surface registration*, Technical Report, 2004.

[17] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Speeded up detection of squared fiducial markers," *Image and vision Computing*, vol. 76, pp. 38–47, 2018.