DRC Team NimbRo Rescue: Perception and Control for Centaur-Like Mobile Manipulation Robot Momaro

Max Schwarz, Marius Beul, David Droeschel, Tobias Klamt, Christian Lenz, Dmytro Pavlichenko, Tobias Rodehutskors, Michael Schreiber, Nikita Araslanov, Ivan Ivanov, Jan Razlaw, Sebastian Schüller, David Schwarz, Angeliki Topalidou-Kyniazopoulou and Sven Behnke

1 Introduction

Disaster scenarios like the Fukushima nuclear accident clearly reveal the need for robots that are capable to meet the requirements arising during operation in real-world, highly unstructured and unpredictable situations, where human workers cannot be deployed due to radiation, danger of collapse or toxic contamination. As a consequence of the incident in Fukushima, the Defense Advanced Research Projects Agency (DARPA) held the DARPA Robotics Challenge[1] (DRC) to foster the development of robots capable of solving tasks which are required to relief catastrophic situations and to benchmark these robots in a competition. During the DRC, the robots needed to tackle eight tasks within one hour: 1. Drive a vehicle to the disaster site, 2. Egress from the vehicle, 3. Open a door, 4. Turn a valve, 5. Cut a hole into a piece of drywall, 6. Solve a surprise manipulation task, 7. Overcome rough terrain or a field of debris, and 8. Climb some stairs. To address this large variety of tasks, we constructed the mobile manipulation robot Momaro and an accompanying teleoperation station for it.

Momaro (see Fig. 1) is equipped with four articulated compliant legs that end in pairs of directly driven, steerable wheels. This unique base design combines

[1] http://archive.darpa.mil/roboticschallenge/.

A version of this article was previously published in the Journal of Field Robotics, vol. 34, issue 2, pp. 400–425, ©Wiley 2017.

M. Schwarz (✉) · M. Beul · D. Droeschel · T. Klamt · C. Lenz · D. Pavlichenko
T. Rodehutskors · M. Schreiber · N. Araslanov · I. Ivanov · J. Razlaw · S. Schüller
D. Schwarz · S. Behnke
Autonomous Intelligent Systems Group, University of Bonn, Bonn, Germany
e-mail: max.schwarz@ais.uni-bonn.de

A. Topalidou-Kyniazopoulou
Centre for Research & Technology - Hellas, Thessaloniki, Greece

M. Spenko et al. (eds.), *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*, Springer Tracts in Advanced Robotics 121,
https://doi.org/10.1007/978-3-319-74666-1_5

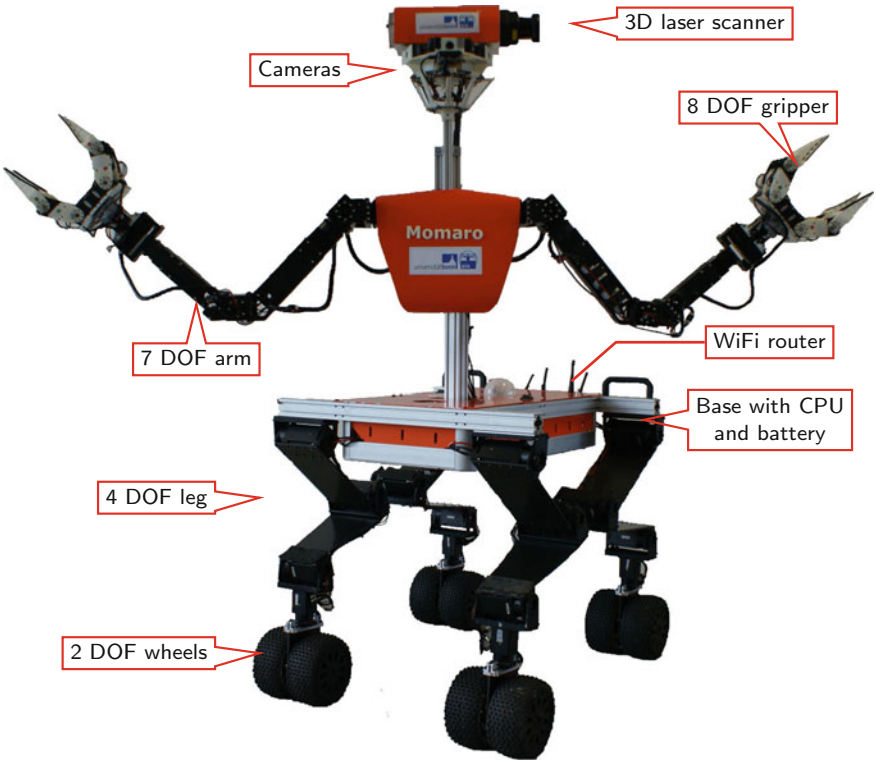

Fig. 1 The mobile manipulation robot Momaro

advantages of driving and stepping locomotion. Wheeled systems, which include also tank-like tracked vehicles, are robust and facilitate fast planning, while being limited in the height differences or terrain types they can overcome. Legged systems require more effort to control and maintain stability, but can cope with quite difficult terrain, because they require only isolated safe footholds. On the downside, they often move slower than wheeled systems. Hybrid systems with a combination of legs and wheels, namely legs ending in wheels, promise to combine the benefits of both locomotion modes. On sufficiently smooth terrain, locomotion is done by driving omnidirectionally on the wheels while adapting to slow terrain height changes with the legs. If larger obstacles prevent driving, the robot switches to stepping locomotion. With these advantages in mind, we chose a hybrid locomotion scheme for Momaro.

To perform a wide range of manipulation tasks, Momaro has an anthropomorphic upper body with two 7 degrees of freedom (DOF) manipulators that end in dexterous grippers. This allows for the single-handed manipulation of smaller objects, as well as for two-armed manipulation of larger objects and the use of tools. Through adjustable base height and attitude and a yaw joint in the spine, Momaro has a work space equal to the one of an adult person.

The DRC requirements are beyond the state of the art of autonomous robotics. As fully autonomous systems which work in these complex environments are not feasible yet, often human intelligence is embedded into the robot through teleoperation to improve the overall performance of the system. Human operators can easily react to unforeseen events, but require awareness of the situation. To this end, we equipped our robot with a 3D laser scanner, multiple cameras, and other sensors.

For effective teleoperation of the many DOF of our robot, intuitive and flexible user interfaces are key. For driving the car, multiple cameras and the visualization of the 3D scene provide good situation awareness and the operator can control the car directly using a steering wheel and a gas pedal. The motions of these remote controllers are mapped to robot limbs actuating the corresponding car controllers. Omnidirectional driving is controlled using a three-axis joystick, based on camera and 3D scene feedback. The velocity commands are mapped to the directions and speeds of the eight robot wheels. To solve complex bimanual manipulation tasks, we developed a teleoperation interface consisting of a stereoscopic head-mounted display (HMD) and two 6D magnetic trackers for the hands of the operator. The operator head motions are tracked to render views based on the available 3D point clouds for the HMD, which follow his motions with low latency. The position and orientation of the magnetic trackers are mapped to the end-effectors of our robot using inverse kinematics with redundancy resolution to calculate positional control commands for Momaro's anthropomorphic arms. For the indoor tasks 4–7, DARPA degraded the communication between the operators and the robot, and data transmission had to be carefully managed. To address this communication restriction, we developed a method for combining a low-latency low-bandwidth channel with a high-latency high-bandwidth channel to provide the operators high-quality low-latency situation awareness.

All the developed components were integrated to a complete disaster-response system, which performed very well at the DARPA Robotics Challenge. Through Momaro, our team NimbRo Rescue solved seven of the eight DRC tasks in only 34 min, coming in as best European team at the 4th place overall. We report in detail on how the tasks were solved. The system was also tested in the DLR SpaceBot Cup and in lab experiments. Our DRC developments led to multiple contributions, which are summarized in this article, including the unique hybrid locomotion concept, good situation awareness despite degraded communication, and intuitive teleoperation interfaces for solving complex locomotion and manipulation tasks. We also discuss lessons learned from the challenging robot operations.

2 Related Work

The need of mobile manipulation has been addressed in the past with the development of a variety of mobile manipulation systems consisting of robotic arms installed on mobile bases with the mobility provided by wheels, tracks, or leg mechanisms. Several research projects exist which use purely wheeled locomotion for their robots

(Mehling et al. 2007; Borst et al. 2009). In previous work, we developed NimbRo Explorer (Stückler et al. 2015), a six-wheeled robot equipped with a 7 DOF arm designed for mobile manipulation in rough terrain encountered in planetary exploration.

Wheeled rovers provide optimal solutions for well-structured, and relatively flat environments, however, outside of these types of terrains, their mobility quickly reaches its limits. Often they can only overcome obstacles smaller than the size of their wheels. Compared to wheeled robots, legged robots are more complex to design, build, and control (Raibert et al. 2008; Roennau et al. 2010; Semini et al. 2011; Johnson et al. 2015) but they have obvious mobility advantages when operating in unstructured terrains and environments. Some research groups have started investigating mobile robot designs which combine the advantages of both legged and wheeled locomotion, using different coupling mechanisms between the wheels and legs (Adachi et al. 1999; Endo and Hirose 2000; Halme et al. 2003).

Recently, the DRC accelerated the development of new mobile manipulation platforms aimed to address disaster response tasks and search and rescue (SAR) operations. While the majority of the teams participating in the DRC Finals designed purely bipedal robots,[2] four of the five best placed teams chose to combine legged with wheeled locomotion, which might indicate advantages of this design approach for the challenge tasks. On the one hand, these robots can move fast over flat terrain using their wheels, on the other hand, they are able to overcome more complex terrain using stepping.

DRC-HUBO of the winning team KAIST is a humanoid robot (Cho et al. 2011; Kim and Oh 2010) capable of bipedal walking. Its powerful joint motors are equipped with an air cooling system to dispense heat efficiently and allow high payloads. DRC-HUBO can rotate its upper body by 180° which enables it to climb a ladder with the knees extending backwards (Lim and Oh 2015). DRC-HUBO is also able to drive over flat terrain, using wheels which are attached to its knees and ankles. To switch between walking and driving, DRC-HUBO transforms between a standing posture and a kneeling posture.

Team IHMC (Johnson et al. 2015) came in second at the DRC Finals and, from the five best placed teams, was the only team using a purely bipedal robot with no additional wheels or tracks: the Atlas robot developed by Boston Dynamics.

CHIMP (Stentz et al. 2015), which placed 3rd in the DRC Finals, was designed to maintain static stability—avoiding engineering challenges that arise if complex balancing control techniques are needed to maintain dynamic stability. The roughly anthropomorphic robot is equipped with powered tracks on its arms and legs, which can be used to drive over uneven terrain. During manipulation tasks, CHIMP rests on the two tracks of its hind legs, which still provide mobility on flat terrain. Raising its frontal limbs allows the robot to use its grippers to manipulate objects. In contrast to our concept, CHIMP does not execute any stepping motions to overcome bigger obstacles like stairs, but instead drives over them on its four tracks while maintaining a low center of mass (COM). The user interface of CHIMP combines manual and

[2] http://archive.darpa.mil/roboticschallenge/teams.html.

autonomous control, for example by previewing candidate free-space motions to the operator.

Likewise, RoboSimian is a statically stable quadrupedal robot with an ape-like morphology (Satzinger et al. 2014; Hebert et al. 2015). It is equipped with four generalized limbs, which can be used for locomotion and manipulation, consisting of seven joints each. All of these 28 joints are driven by identical actuators to ease development and maintenance of the robot hardware. Furthermore, it is equipped with under-actuated hands at the end of its limbs with fewer fingers and active DOF than a human hand. Besides executing stepping motions with its limbs, it is also capable of driving on four wheels. For this purpose, RoboSimian can lower itself onto two active wheels attached to its trunk and two caster wheels on two of its limbs. This allows the robot to drive on even terrain, while still being able to manipulate objects using its other two limbs. RoboSimian placed 5th in the competition.

In contrast to DRC-HUBO, CHIMP, and RoboSimian, our robot Momaro is capable of driving omnidirectionally, which simplifies navigation in restricted spaces and allows us to make small positional corrections faster. Furthermore, our robot is equipped with six limbs, two of which are exclusively used for manipulation. The use of four legs for locomotion provides a large and flexible support polygon when the robot is performing mobile manipulation tasks.

We developed a telemanipulation interface for our robot using an immersive 3D HMD (Oculus Rift) and two 6D controllers (Razer Hydra), allowing an operator to intuitively manipulate objects in the environment. Telemanipulation interfaces using 3D perception and a HMD have been addressed by multiple groups, for example for SAR robots (Martins and Ventura 2009), explosive ordnance disposal (Kron et al. 2004), or in surgery (Ballantyne and Moll 2003; Hagn et al. 2010). In contrast, our telemanipulation solution consists of low-cost consumer-grade equipment.

The idea of using consumer-grade equipment for robotic applications is not new. Kot and Novák (2014) used the Oculus Rift as well in their mobile manipulation setup using a four-wheeled robot with a 3 DOF arm. Similarly, Smith and Christensen et al. (2009) used the low-priced Wiimote game controller with an additional IR camera to track the position and orientation of the operator hand. They use a minimum jerk human motion model to improve the precision of the tracking and achieved good results for minimally instructed users in a simple manipulation task. In contrast to the Wiimote, which can only measure linear accelerations, the Razer Hydra is able to determine absolute positions using a magnetic field. Compared to the previous work on telemanipulation, we describe a system that can be intuitively teleoperated by a human operator—even under degraded network communication—and is highly mobile by using a combination of legged and wheeled locomotion.

3 Mobile Manipulation Robot Momaro

Our mobile manipulation robot Momaro (see Fig. 1) was specifically designed for the requirements of the DRC. Besides the overall goal to solve all DRC tasks, we specified

additional design constraints: A *bimanual design* offers both the ability to perform complex or strenuous manipulation tasks which might be impossible using only one hand, and also adds redundancy for one-handed tasks. Bimanual manipulation is also a long-standing interest of our research group, particularly in context of service robotics (Stückler et al. 2014). A *large support polygon* minimizes the need for balance control, which might be challenging, e.g., for bipedal robots. *Legs* offer the ability to step over or climb on obstacles. A *lightweight robot* is less dangerous and also easier to handle than heavy robots requiring special moving equipment. The capability of *omnidirectional movement* allows faster and more precise correction movements in front of manipulation tasks, when compared to, e.g., a robot that needs to turn in order to move sideways. Finally, since our hardware engineering capacities were limited, we wanted to use *off-the-shelf components* as much as possible.

3.1 Kinematic Design

Driven by the off-the-shelf and lightweight design goals, we decided to power all robot joints by Robotis Dynamixel actuators (see Table 1), which offer a good torque-to-weight ratio. Notably, all other high-placed DRC designs use custom actuator designs. Figure 3 gives an overview of the kinematic structure of Momaro.

Since state of the art approaches for bipedal locomotion on terrain are prone to falls and current generation robots are mostly not able to recover after these falls by themselves, we decided to equip Momaro with a total of four legs to minimize the probability of falling. As robot locomotion using stepping is comparably slow, the legs end in pairs of steerable wheels. The legs have three pitch joints in hip, knee and ankle, allowing the adjustment of the wheel pair position relative to the trunk in the sagittal plane. Furthermore, the ankle can rotate around the yaw axis and the two wheels can be driven independently. This allows the robot to drive omnidirectionally on suitable terrain, while also stepping over obstacles too high to drive over.

The leg segments are carbon fiber springs, thus providing passive adaptation to terrain. The foreleg extension varies 40 cm from minimum to maximum, i.e. from lowest to highest configuration of the robot. In the minimum configuration, Momaro has a chassis clearance of 32 cm. The hind legs can extend 15 cm more to allow the robot to climb steeper inclines while keeping the basis level. While the legs can be used for locomotion, they also extend the workspace of the robot for manipulation tasks, e.g., by changing the height of the base or by pitching/rolling the base through antagonistic leg length changes. The wheels are soft foam-filled rubber wheels, which provide ample traction. Their radius of 8 cm and the flexible suspension formed by the carbon fiber springs allows the robot to ignore most obstacles lower than approximately 5 cm. Since our manipulation interfaces (see Sect. 7) do not require precise base positioning, the spring design does not decrease manipulation capabilities. Additionally, unintended base movement is measured using the built-in IMU and compensated for during sensor data processing (see Sect. 4).

Table 1 Robotis Dynamixel actuator models used in Momaro

Joint	Model	Mass (g)	Torque (Nm)
Hip	H54-200-S500-R	855	44.2
Knee	H54-200-S500-R	855	44.2
Ankle (pitch)	H54-100-S500-R	732	24.8
Ankle (yaw)	H42-20-S300-R	340	6.3
Wheels	2× H42-20-S300-R	340	6.3
Torso (yaw)	H42-20-S300-R	340	6.3
Laser	MX-64	126	6.0
Shoulder (r.+p.)	2× H54-200-S500-R	855	44.2
Shoulder (yaw)	H54-100-S500-R	732	24.8
Elbow	H54-100-S500-R	732	24.8
Wrist (roll)	H42-20-S500-R	340	6.3
Wrist (pitch)	H42-20-S300-R	340	6.3
Wrist (yaw)	L42-10-S300-R	257	1.4
Proximal fingers	4× MX-106	153	8.4
Distal fingers	4× MX-64	126	6.0

The colors match the actuator colors in Figs. 2 and 3.

On top of its flexible base, Momaro has an anthropomorphic upper body consisting of two adult-sized, 7 DOF arms (see Figs. 1 and 3) and a sensor head. The upper body of the robot is connected to the base by a torso yaw joint that increases the workspace of the end-effectors and allows the system to execute more tasks without the use of locomotion. Each arm ends in a custom hand equipped with four 2 DOF fingers (see Fig. 2). While the proximal segment of each finger is rigid, Festo FinGrippers are used as distal segments. These grippers deform if force is applied to them to better enclose a grasped object by enlarging the contact surface between object and gripper. The position of the finger tips on each finger can manually be reconfigured to allow pinch grips as well as cylindrical grasps.

Momaro is relatively lightweight (58 kg) and compact (base footprint 80 cm × 70 cm), which means that it can be carried comfortably by two people, compared to larger crews and equipment like gantries needed to carry other robots of comparable size. Since the legs and upper body can be detached, the robot can be transported in standard suitcases.

3.2 Sensing

Momaro's main sensor for environmental perception is a 3D rotating laser scanner on its sensor head (see Fig. 4). It consists of a Robotis Dynamixel MX-64 actuator, which rotates a Hokuyo UTM-30LX-EW laser scanner around the vertical axis. A PIXHAWK IMU is mounted close to the laser scanner, which is used for motion compensation during scan aggregation and state estimation. Three Full HD color cameras are also attached to the sensor head for a panoramic view of the environment

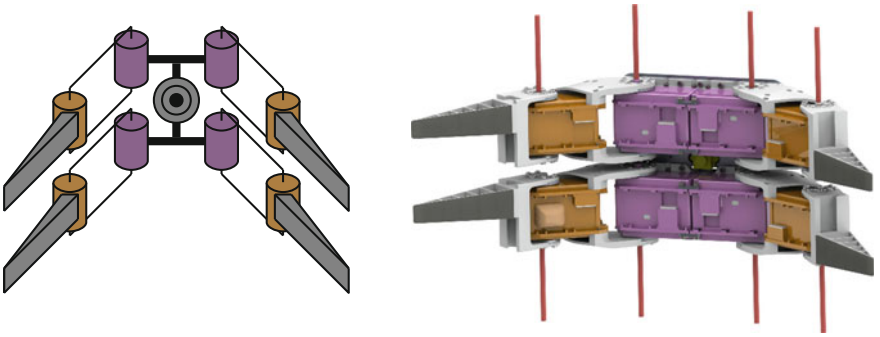

Fig. 2 Gripper design. Left: Kinematic tree of one of Momaro's hands. While all segments connecting the joints are rigid, the distal finger segments deform if force is applied to them. Proportions are not to scale. The color camera mounted in the hand is visible in the center. Right: CAD rendering of the hand. The finger joint axes are marked with red lines

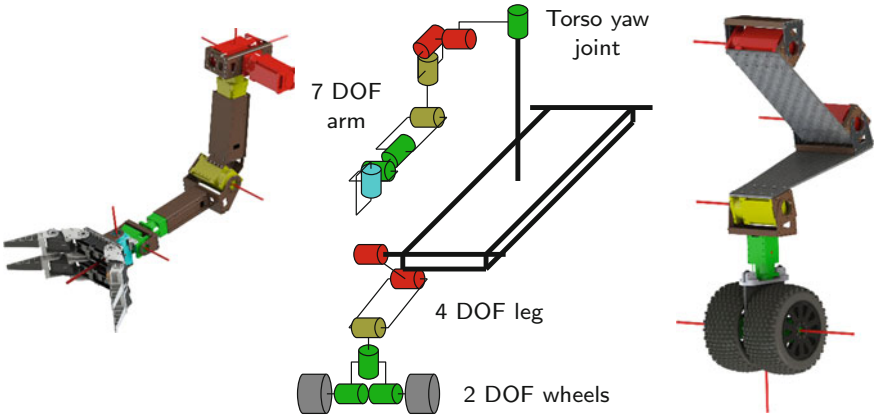

Fig. 3 Kinematic layout. Left: CAD rendering of the right arm. Joint axes are marked with red lines. Center: Kinematic tree. For clarity, the figure only shows a part of the robot and does not show the hand with its additional eight DOF. Proportions are not to scale. Right: CAD rendering of the front right leg. The six joint axes in hip, knee, ankle pitch, ankle yaw, and wheels are marked with red lines

in front of the robot and a top-down wide angle camera is used to observe the movement of the arms of the robot and its interaction with the environment. Each hand is equipped with a camera which is located between its fingers. These cameras can be used to visually verify the correct grasp of objects. Furthermore, since these cameras are mounted at the end-effectors of the robot and can therefore be moved, they can be used to extend the view of the operators, for example, to view a scene from another perspective if the view from the head mounted top-down camera is occluded. Finally, the robot also carries a downward-facing wide-angle camera under its base which allows the operators to monitor the wheels and the surface beneath Momaro.

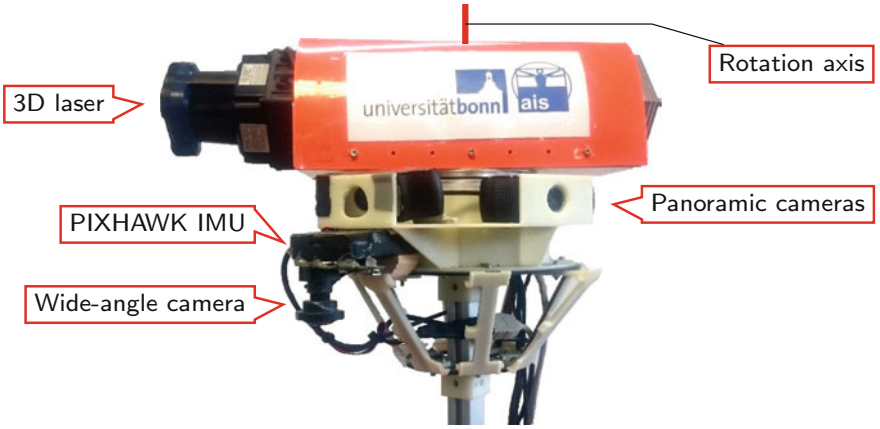

Fig. 4 Sensor head carrying 3D laser scanner, IMU, and panoramic cameras

Since the right hand is used for the more complex tasks, it is equipped with additional sensors. A microphone connected to the hand camera can be used for auditory feedback to the operators. Underneath the hand, we mounted an infrared distance sensor to measure distances within the environment.

3.3 Electronics

Figure 5 shows an overview over the electrical components of Momaro. In its base, Momaro carries an on-board computer with a fast CPU (Intel Core i7-4790K @4–4.4 GHz) and 32 GB RAM. For communication with the operator station, it is equipped with a NETGEAR Nighthawk AC1900 WiFi router, which allows 2.4 GHz and 5 GHz transmission with up to 1300 Mbit/s. We make use of a total of six (one for each leg and arm) Crumb2560 microcontroller boards, which bridge high-level USB commands from the computer to low-level servo control commands and vice versa. Performance of the joint actuators is continuously monitored. Feedback information includes measured position, applied torque, and actuator temperature. Like the microcontroller boards, all cameras, the servo for rotation of the laser, and the PIXHAWK IMU are connected via USB 2.0 for a total of 16 USB devices. The laser scanner is connected via 100 Mbit/s LAN through a slip ring.

In case of undesirable actions or emergencies, Momaro can be emergency-stopped through two emergency stop switches. One is mounted on the base of the robot for easy access during development, the other one is the wireless E-Stop system mandatory for all DRC competitors. The E-stops are connected to the actuator control microcontrollers. If the robot is E-stopped, it stops all currently active servo commands.

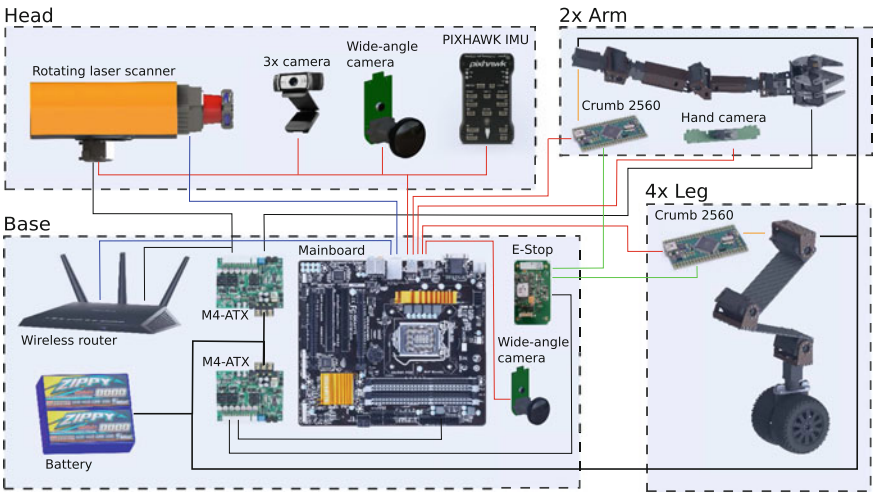

Fig. 5 Simplified electrical schematics of Momaro. We show USB 2.0 data connections (red), LAN connections (blue), E-Stop related wiring (green), the low-level servo bus system (orange), and power connections (black). Thick black lines indicate battery power, whilst thin black lines carry 12 V

Power is supplied to the robot by a six-cell LiPo battery with 16 Ah capacity at 22.2 V nominal voltage, which yields around 1.5–2 h run time, depending on the performed tasks. Batteries are hot-swappable and thus can be easily exchanged while running. For comfortable development and debugging, they can also be substituted by a power supply.

4 Perception

To assist the operators in navigation and manipulation tasks, we construct a 3D egocentric local multiresolution grid map by accumulating laser range measurements that are made in all spherical directions. The architecture of our perception and mapping system is outlined in Fig. 6. 3D scans are acquired in each full rotation of the laser. Since a rotation takes time, the motion of the robot needs to be compensated when assembling the scan measurements into 3D scans (Sect. 4.1). We first register newly acquired 3D scans with the so far accumulated map and then update the map with the registered 3D scan to estimate the motion of the robot, compensating for drift of the wheel odometry and IMU measurements.

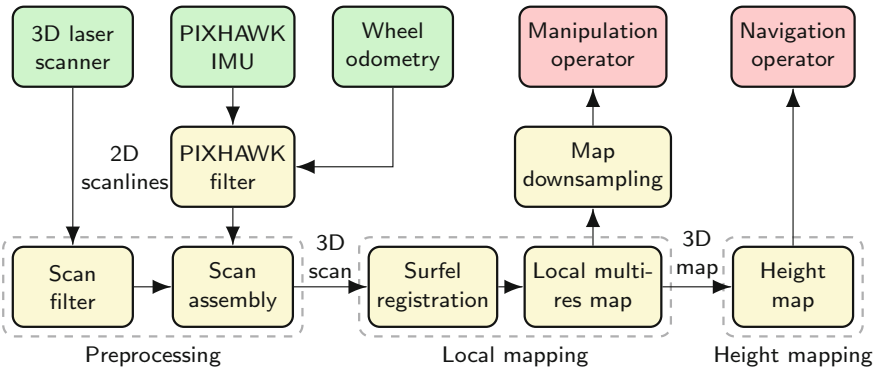

Fig. 6 Overview of our 3D laser perception system. The measurements are processed in preprocessing steps described in Sect. 4.1. The resulting 3D point cloud is used to estimate the transformation between the current scan and the map. Registered scans are stored in a local multiresolution map

4.1 Preprocessing and 3D Scan Assembly

The raw measurements from the laser scanner are subject to spurious measurements at occluded transitions between two objects. These so-called *jump edges* are filtered by comparing the angle of neighboring measurements. After filtering for jump edges, we assemble a 3D scan from the 2D scans of a complete rotation of the scanner. Since the sensor is moving during acquisition, we undistort the individual 2D scans in two steps.

First, measurements of individual 2D scans are undistorted with regards to the rotation of the 2D laser scanner around the sensor rotation axis. Using spherical linear interpolation, the rotation between the acquisitions of two scan lines is distributed over the measurements.

Second, the motion of the robot during acquisition of a full 3D scan is compensated. Due to Momaro's flexible legs, it is not sufficient to simply use wheel odometry to compensate for the robot motion. Instead, we estimate the full 6D state with the PIXHAWK IMU attached to Momaro's head. Here we calculate a 3D attitude estimate from accelerometers and gyroscopes to compensate for rotational motions of the robot. Afterwards, we filter the wheel odometry with measured linear acceleration to compensate for linear motions. The resulting 6D state estimate includes otherwise unobservable motions due to external forces like rough terrain, contacts with the environment, wind, etc. It is used to assemble the individual 2D scans of each rotation to a 3D scan.

4.2 Local Multiresolution Map

The assembled 3D scans are accumulated in a hybrid local multiresolution grid-based map. Measurements and occupancy information are stored in grid cells that increase in size with the distance from the robot center. The individual measurements are stored in ring buffers enabling constant size in memory. More recent measurements replace older measurements. By using multiresolution, we gain a high measurement density in the close proximity to the sensor and a lower measurement density far away from our robot, which correlates with the sensor characteristics in relative distance accuracy and measurement density. Compared to uniform grid-based maps, multiresolution leads to the use of fewer grid cells, without losing relevant information and consequently results in lower computational costs. Figure 7 shows an example of our grid-based map.

Maintaining the egocentric property of the map necessitates efficient map management for translation and rotation during motion. Therefore, individual grid cells are stored in ring buffers to allow shifting of elements in constant time. Multiple ring buffers are interlaced to obtain a map with three dimensions. In case of a translation of the robot, the ring buffers are shifted whenever necessary. For sub-cell-length translations, the translational parts are accumulated and shifted if they exceed the length of a cell.

Newly acquired 3D scans are aligned to the local multiresolution map by our surfel registration method (Droeschel et al. 2014). We gain efficiency by summarizing individual points in each grid cell by a sample mean and covariance.

4.3 Height Mapping

Besides assisting the operators for navigation and manipulation tasks, the local map is used by the autonomous stepping module to plan footsteps. To this end, the 3D map

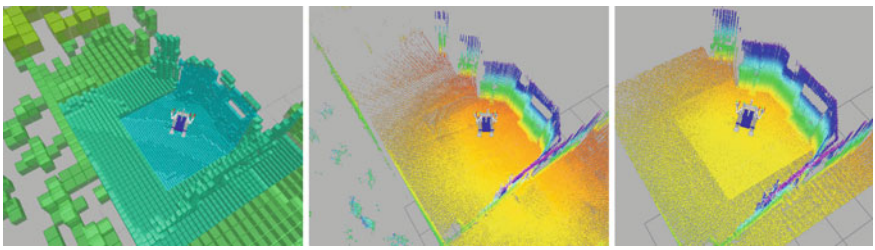

Fig. 7 The local multiresolution grid-based map during the first DRC competition run. Left: The grid-based local multiresolution map. Cell size (indicated by color) increases with the distance from the robot. Middle: 3D points stored in the map on the robot. Right: Downsampled and clipped local map, transmitted to the operator for manipulation and navigation tasks. Color encodes height above ground

is projected into a 2.5D height map, shown in Fig. 13. Gaps in the height map (cells without measurements) are filled with the local minimum if they are within a distance threshold of valid measurements (10 cm in our experiments). The rationale for using the local minimum is that gaps in the height map are usually caused by occlusions. The high mounting position of the laser on the robot means that low terrain is more likely occluded than high terrain. The local minimum is therefore a good guess of missing terrain height. After filling gaps in the height map, the height values are filtered using the fast median filter approximation using local histograms (Huang et al. 1979). The filtered height map is suitable for planning footsteps.

5 Communication

One constraint during the DRC was the limited communication between the operator station and the robot, which was enforced to simulate degenerated communication as may occur in a real-world mission. The uplink from the operator station to the robot was limited to 9600 bit/s at all times. The downlink from the robot to the operator station was limited to 300 Mbit/s outside of the building during the driving tasks, the door task, and the stairs task. Inside the building (defined by the door thresholds), the downlink was limited to 9600 bit/s, interleaved with one second long bursts of 300 Mbit/s bandwidth. These burst became more frequent during the run and the blackouts vanished completely after 45 min into the run. As usual, the wireless communication link does not guarantee packet delivery, so communication systems had to deal with packet loss.

To cope with this degraded communication, sensor information cannot be transferred unselected and uncompressed. The main idea of our communication system is to transfer stationary information about the environment over the high-latency high bandwidth channel, while we use the low-latency low bandwidth channel to transfer frequently changing data. Both are then combined on the operator station to render immersive 3D visualizations with low latency for the operators.

5.1 Communication Architecture

Our communication architecture is shown in Fig. 8. The main topology was formed by the DARPA requirements, which placed the Degraded Communications Emulator (DCE) between the operator crew and the robotic system. To allow buffering and relaying of data over the high-bandwidth link, we make use of the option to include a separate field computer, which is connected via Ethernet to the DCE on the robot side. The key motivation here is that the wireless link to the robot is unreliable, but unlimited in bandwidth, while the link over the DCE is reliable, but limited in bandwidth. Placing the field computer directly after the DCE allows exploitation of the characteristics of both links.

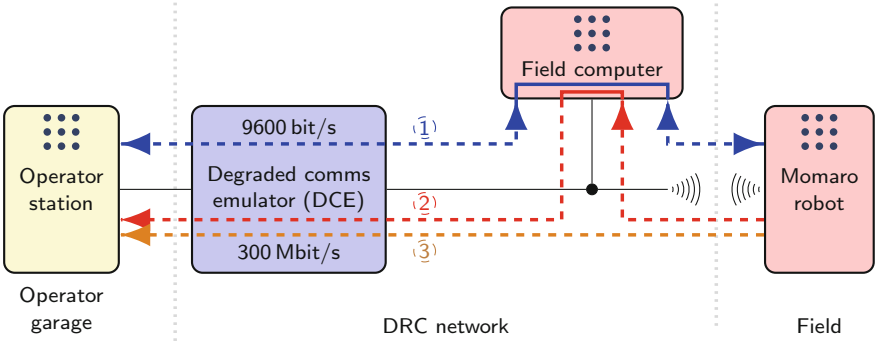

Fig. 8 Communication architecture. Components in the vicinity of the operators are shown in yellow, DARPA-provided components in blue, components in the "field"-network in red. Solid black lines represent physical network connections. Dashed lines show the different channels, which stream data over the network (blue (1): low bandwidth, red (2): bursts, brown (3): direct imagery). The ROS logo indicates a ROS master

On the operator side of the DCE, the operator station is the central unit. Since our operator crew consists of more than one person, we have the option to connect multiple specialist's notebooks to the operator station. Finally, the computer running our telemanipulation interface (see Sect. 7.3) is also directly connected to the operator station. Since we use the ROS middleware for all software components, separate ROS masters run on the robot, the field computer, and the operator station. The communication between these masters can be split into three channels, which will be explained below.

5.1.1 Low-Bandwidth Channel

The low-bandwidth channel is a bidirectional channel between the operator station and robot (blue (1) in Fig. 8). It uses the low-bandwidth link of the DCE and is therefore always available. Since the bandwidth is very limited, we do most compression on the field computer, where we can be sure that packets sent to the operator station are not dropped, which would waste bandwidth.

Since the low-bandwidth link over the DCE was the main live telemetry source for the operator crew, we spent considerable effort on compressing the data sent over this link in order to maximize the amount of information the system provides. The transmitter running on the field computer sends exactly one UDP packet per second. The bandwidth is thus easily controlled by limiting the UDP payload size. Since the amount of data is much less in the other direction, the transmitter on the operator station sends operator commands with up to 5 Hz. Payload sizes in bits are given in Table 2.

Table 2 Average bit rates of topics transmitted over the low-bandwidth link

Robot → Operator			Operator → Robot		
Channel/Topic	Rate (Hz)	Avg. Bit/message	Channel/Topic	Rate (Hz)	Avg. Bit/message
H.264 Camera image	1	6000	Arm control	5	96
Joint positions	1	736	Joystick command	5	56
Base status	1	472	Generic motion[a]	-	144
3D Contour points	1	250	Motion play request	-	80
Transforms	1	136			
Audio amplitude	1	8			
Sum per 1s		7602			-[b]

Topics with rate of "-" are transmitted only on operator request
[a]Generic transport for all kinds of keyframe motions. Here: one frame using Cartesian EEF pose
[b]Summation is not applicable here, since the total bit rate depends heavily on operator action

For low-level compression of floating point numbers as well as 3D/4D vectors and quaternions, we developed a small helper library, which is freely available.[3] It employs techniques originally developed for compressing geometry data for transfers between CPU and GPU: Quaternions are compressed using the hemi-oct encoding (Cigolle et al. 2014), while 3D vectors are compressed using a face-centered cubic packing lattice. The lattice approach offers better average discretization error than naive methods which discretize each axis independently.

Since visual information is of crucial importance to human operators, we also transmit a low resolution video stream. As Momaro is equipped with a variety of cameras, an operator needs to select the camera whose output should be sent over the low bandwidth link. The selection of the camera depends on the currently executed task and is also often changed during a task. Note that all camera images are also transmitted over the high-bandwidth link. The purpose of low-bandwidth imagery is merely to provide low-latency feedback to the operators. The selected camera image is compressed at the field PC using the H.264 codec. Before compression, the image is downscaled to 160×120 pixels. Furthermore, we use the periodic intra refresh technique instead of dedicated keyframes, which allows to specify a hard packet size limit for each frame. While the compression definitely reduces details (see Fig. 9), the camera images still allow the operators to make fast decisions without waiting for the next high-bandwidth burst image.

Measured joint positions are discretized and transmitted as 16 bit integers (8 bit for the most distal joints in the kinematic tree). The joint positions are used for forward kinematics on the operator station to reconstruct poses of all robot parts. A small

[3]https://github.com/AIS-Bonn/vector_compression.

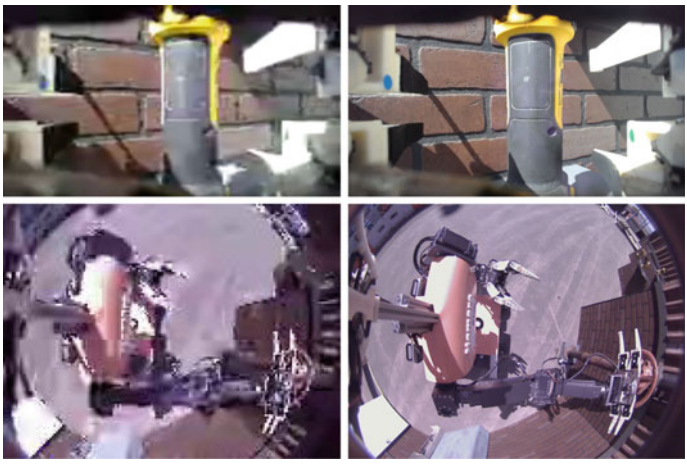

Fig. 9 Comparison of webcam images over low- and high-bandwidth channels. The top images were captured by the right hand camera, looking at the drill tool. The bottom images show the overhead view while the robot is grasping the valve

number of 3D rigid body transformations are sent over the network, including the current localization pose, odometry, and IMU information. The transforms are sent as 3D vector and quaternion pairs, compressed using the library mentioned above.

Up to 125 3D *contour points* are compressed and sent to the operator for display. These contour points are extracted from the laser scans and are meant to outline the contour of the endeffector and objects in its direct vicinity. By transmitting contour points over the low-bandwidth channel, the operator is provided with live sensory feedback from the laser scanner during a manipulation task. Figure 10 shows the extracted contour points from a typical manipulation task. In order to minimize the number of points that are transmitted, we detect measurements on the manipulator and the close-by object by applying a combination of filters on the raw laser scans in a given scan window extracted from the last three 2D laser scans. First, so-called *jump edges*—occurring at occluded transitions between two objects—are removed by filtering neighboring measurements by their angle. Then, we detect edge points by applying a Sobel edge filter on the distance measurements in a scan window. To account for edges resulting from noisy measurements, distance measurements are smoothed by a Median filter before applying the Sobel filter. Since dull or curvy edges may result in numerous connected edge points, we further reduce the remaining edge points by applying a line segment filter. The line segment filter reduces a segment of connected edge points to its start and end point. The corresponding 3D points of the remaining distance measurements are transmitted to the operator as contour points. Selecting contour points by filtering the distance measurements of the raw laser scans—contrary to the detection in 3D point clouds—results in a robust and efficient

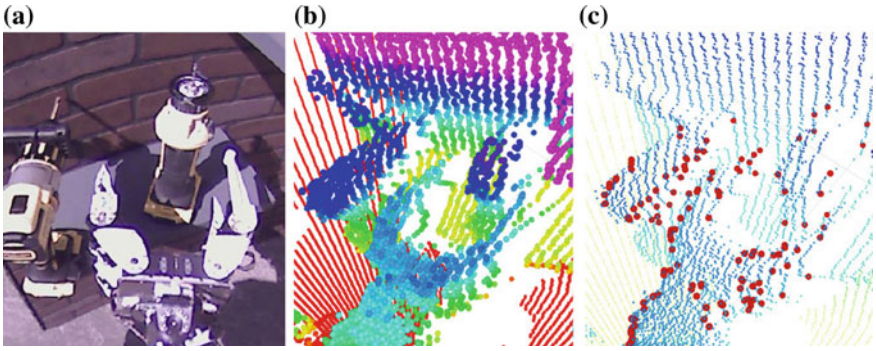

Fig. 10 3D contour points for a typical manipulation task (grasping the drill). **a** The overhead camera image. **b** The raw laser scans (color encodes height from ground). **c** The resulting contour points (red)

detector which allows us to transmit live sensory feedback over the low-bandwidth channel.

Telemetry from the robot base includes the current support polygon, estimated COM position, emergency stop status, infrared distance measurement from the hand, and the maximum servo temperature. Finally, the low-bandwidth link also includes the measured audio amplitude of the right hand camera microphone, which allows us to easily determine whether we succeeded in turning the drill on.

5.1.2 High-Bandwidth Burst Channel

Since the connection between robot and field PC is always present, irrespective of whether the DCE communication window is currently open, we use this connection (red (2) in Fig. 8) to transfer larger amounts of data to the field PC for buffering.

During our participation in the DLR SpaceBot Cup (Stückler et al. 2015), we developed a robust software module for communication between multiple ROS masters over unreliable and high-latency networks. This module was extended with additional features during the DRC and is now freely available under BSD-3 license.[4] It provides transport of ROS topics and services over TCP and UDP protocols. Since it does not need any configuration/discovery handshake, it is ideally suited for situations where the connection drops and recovers unexpectedly. The high-bandwidth channel makes exclusive use of this *nimbro_network* software. This made fast development possible, as topics can be added on-the-fly in configuration files without developing specific transport protocols. After DRC, additional improvements to *nimbro_network* have been made, e.g., adding forward error correction for coping with large packet loss ratios.

[4] https://github.com/AIS-Bonn/nimbro_network.

The transmitted ROS messages are buffered on the field computer. The field computer sends a constant 200 MBit/s stream of the latest received ROS messages to the operator station. This maximizes the probability of receiving complete messages during the short high-bandwidth communication windows inside the building.

The transferred data includes:

- JPEG-compressed camera images from all seven cameras on board, plus two high-resolution cut-outs of the overhead camera showing the hands,
- compressed[5] point cloud from the ego-centric 3D map (see Sect. 4),
- ROS log messages,
- servo diagnostics, and
- miscellaneous diagnostic ROS topics.

The 3D data received in the communication bursts is shown to the operators and transformed into a fixed frame using the low-latency transform information received over the low-bandwidth channel.

5.1.3 High-Bandwidth Direct Imagery

During the outside tasks, the high-bandwidth link is always available. This opens the possibility of using streaming codecs for transmitting live imagery, which is not possible in the inside mode, where communication blackouts would corrupt the stream. Thus, an additional high-bandwidth channel using the *nimbro_network* module carries H.264 encoded camera streams of the main overhead camera and the right hand camera. The streams use an increased frame rate of 5 Hz to allow low-latency operator control. These camera streams are used during the drive task for steering the car. The channel is shown in brown (3) in Fig. 8.

6 Control

The Momaro robot is challenging to control because of its hybrid locomotion concept and the many DOF involved. This section describes the control strategies we developed.

6.1 Kinematic Control

The kinematic control implemented in Momaro (see Fig. 11) follows a straightforward approach. All limbs and the torso yaw joint are considered separately.

[5] The point clouds were compressed using the PCL point cloud compression.

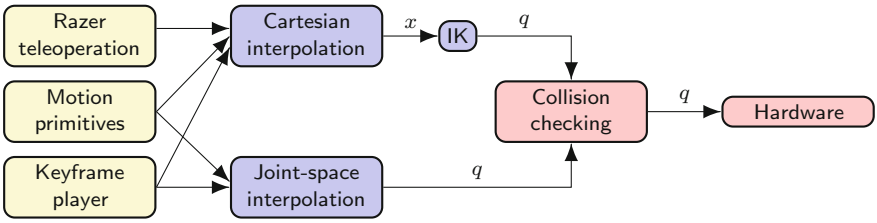

Fig. 11 Kinematic control architecture for one limb. The goal configuration can be specified in joint space or Cartesian space using the magnetic trackers, motion primitives, or the keyframe player. After interpolation (and IK for Cartesian poses x), the resulting joint configuration q is checked for collisions and sent to the hardware

A Cartesian or joint-space goal configuration for a limb is defined through telemanipulation (see Sect. 7) or dedicated motion primitives (e.g., used for the DRC wall cutting task). The Reflexxes library (Kröger 2011) is used to interpolate between the current and desired position in Cartesian or joint space. If concurrent limb motion is desired, Cartesian and joint-space goals can be mixed freely for the different limbs. The interpolation is done such that all limbs arrive at the same time. Interpolated Cartesian poses are converted to joint space positions via inverse kinematics. Finally, the new robot configuration is checked for self-collisions and, if collision-free, fed to the low-level hardware controllers for execution.

For the 7 DOF arms, we calculate the inverse kinematics with redundancy resolution using the selectively damped least squares (SDLS) approach (Buss and Kim 2005). SDLS is an iterative method based on the singular value decomposition of the Jacobian of the current robot configuration. It applies a damping factor for each singular value based on the difficulty of reaching the target position. Furthermore, SDLS sets the target position closer to the current end-effector position if the target position is too far away from the current position. SDLS robustly computes target position as close as possible to 6D poses if they are not within the reachable workspace of the end-effector. Furthermore, we combine SDLS with a nullspace optimization based on the projection of a cost function gradient to the nullspace (Liegeois 1977). The used cost function is a sum of three different components:

1. Joint angles near the limits of the respective joint are punished to avoid joint limits, if possible.
2. The difference between the robot's last and newly calculated configuration is penalized to avoid jumps during a motion.
3. The difference from a user-specified "convenient" configuration and the newly calculated configuration is punished to reward this specific arm position. We chose this convenient configuration to position the elbow of each arm next to the body.

For the legs, the IK problem is solved with a custom analytical kinematics solver. Since the legs have four DOF (excluding the wheels), the solution is always unique as long as it exists.

Calculated joint configurations are checked for self-collisions with simplified convex hull collision shapes using the MoveIt! library.[6] Motion execution is aborted before a collision occurs. The operator can then move the robot out of the colliding state by moving in another direction.

6.2 Omnidirectional Driving

The wheel positions $\mathbf{r}^{(i)}$ relative to the trunk determine the footprint of the robot, but also the orientation and height of the robot trunk. An operator can manipulate the positions via a graphical user interface (see Sect. 7.4) either directly for each wheel by dragging it around, moving all wheels together (thus moving the trunk relative to the wheels) or rotating all wheel positions around the trunk origin (thus controlling the trunk orientation).

An operator can control the base omnidirectional driving using a joystick, which generates a velocity command $\mathbf{w} = (v_x, v_y, \omega)$ with horizontal linear velocity $\mathbf{v}$ and rotational velocity ω around the vertical axis. The velocity command is first transformed into the local velocity at each wheel i:

$$\begin{pmatrix} v_x^{(i)} \\ v_y^{(i)} \\ v_z^{(i)} \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \omega \end{pmatrix} \times \mathbf{r}^{(i)} + \dot{\mathbf{r}}^{(i)}, \tag{1}$$

where $\mathbf{r}^{(i)}$ is the current position of wheel i relative to the base. The kinematic velocity component $\dot{\mathbf{r}}^{(i)}$ allows simultaneous leg movement while driving. Before moving in the desired direction, the wheel pair needs to rotate to the yaw angle $\alpha^{(i)} = \text{atan2}(v_y^{(i)}, v_x^{(i)})$.

After all wheels are properly rotated, each wheel moves with linear velocity $||(v_y^{(i)}, v_x^{(i)})^T||$. While driving, the robot continuously adjusts the orientation of the ankle, using IMU information to keep the ankle yaw axis vertical and thus retains omnidirectional driving capability.

6.3 Semi-autonomous Stepping

In teleoperated scenarios, a suitable balance between autonomous actions conducted by the robot, and operator commands has to be found, due to the many DOF that need to be controlled simultaneously and due to typically limited communication bandwidth. If the terrain is not known before the robotic mission, the motion design approach described above is not applicable. Our system addresses these scenarios by semi-autonomously executing weight shifting and stepping actions when required and requested by an operator. In order to plan footsteps, the autonomous

[6] http://moveit.ros.org.

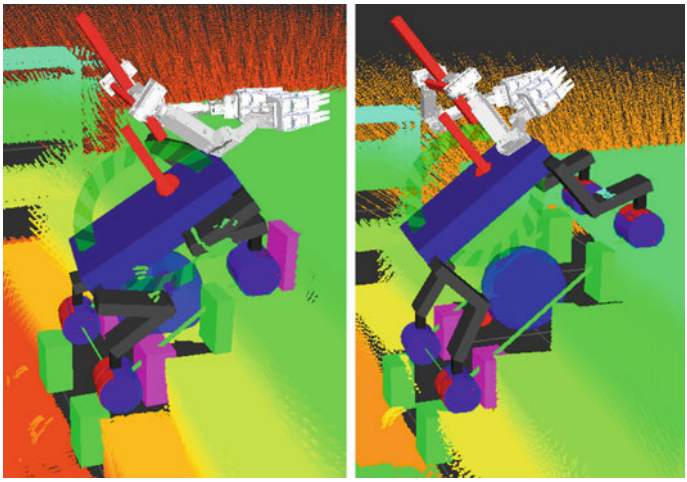

Fig. 12 Momaro climbing stairs in simulation. The purple and green boxes indicate detected obstacles which constrain the wheel motion in forward and backward direction, respectively

stepping module uses the 2.5D height map generated from the 3D laser measurements, described in Sect. 4.3. For details on the approach, see Schwarz et al. (2016b).

While the operator always retains control of the velocity of the robot base using a joystick, steps can be triggered either automatically or manually. The automatic mode always decides on the wheel pair which most urgently needs stepping for continued base movement with the requested velocity. To this end, we detect obstacles along the travel direction of the wheels (see Fig. 12).

To be able to lift a whee, the robot weight must be shifted away from it. Ideally, the 2D projection of the COM of the robot should lie in the center of the triangle formed by the other three wheel pairs (see Fig. 13). This ensures static balance of the robot while stepping. The system has three means for achieving this goal:

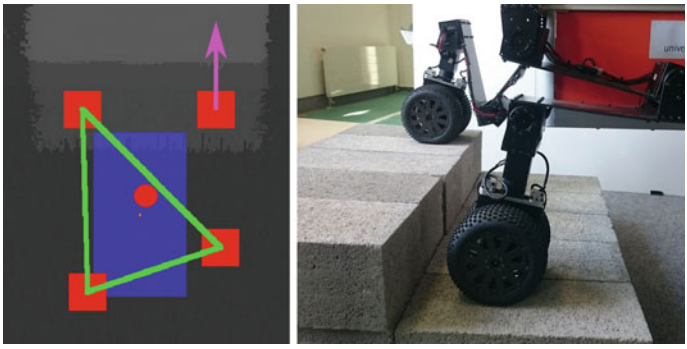

Fig. 13 Left: 2D height map of Momaro standing on two steps of a set of stairs in our lab. The robot is in stable configuration to lift the right front leg. Red rectangles: Wheel positions, red circle: COM, blue: robot base, green: support polygon. Right: The right front leg is lifted and placed on the next step

1. moving the base relative to the wheels in sagittal direction,
2. driving the wheels on the ground relative to the base, and
3. modifying the leg lengths (and thus the base orientation).

All three methods have been used in the situation depicted in Fig. 13. The balance control behavior ensures static balance using foot motions on the ground (constrained by the detected obstacles) and leg lengths. If it is not possible to move the COM to a stable position, the system waits for the operator to adjust the base position or orientation to resolve the situation.

The stepping motion itself is a parametrized motion primitive in Cartesian space. The target wheel position is determined in the height map as the first possible foothold after the height difference which is currently stepped over. As soon as the wheel is in the target position, the weight is shifted back using balance control. The operator is then free to continue with either base velocity commands or further steps.

7 Operator Interface

During DRC runs, we split all operation between the "lower body operator", and the "upper body operator", and a total of seven support operators. One support operator assists the upper body operator by modifying his view. Two operators are responsible for managing the local multiresolution map by clearing undesirable artifacts or highlighting parts of the map for the upper body operator. Another support operator monitors the hardware and its temperature during the runs. Two more operators assist the upper body operator by triggering additional predefined parameterized motions and grasps and are able to control the arms and grippers in joint space as well as in task space using a graphical user interface if necessary. While the system is designed to be controllable using a minimum of two operators (the lower- and upper-body operators), the actual number of operators is flexible.

7.1 Situational Awareness

The main operator interface shown on the dedicated operator station computer over four screens can be seen in Fig. 14. Operator situational awareness is gained through 3D environment visualization and transmitted camera images. The upper screen shows camera images from the overhead camera, ground camera, and hand cameras. It also shows higher-resolution cut-outs from the overhead camera centered on the hands. For all camera images, the view always shows the last received image, independent of the image source (low-bandwidth or high-bandwidth burst). This ensures that operators always use newest available data.

The lower middle screen shows a 3D visualization of the robot in its environment. Serving as main environmental representation (see Fig. 15), a downsampled and

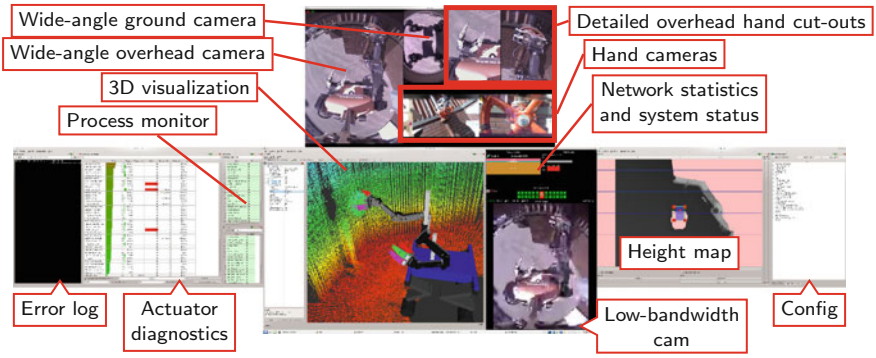

Fig. 14 GUI on the main operator station, during the DRC valve task

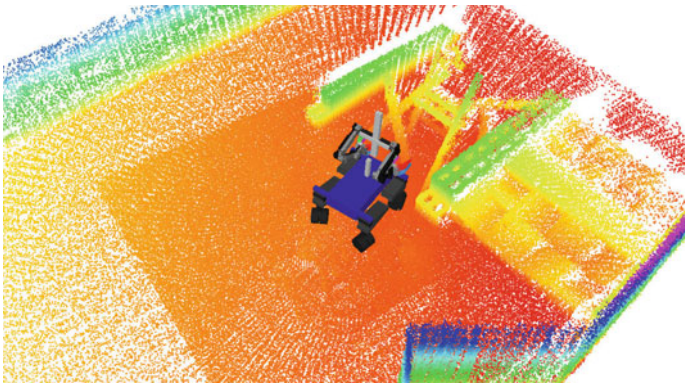

Fig. 15 Point cloud of the egocentric multiresolutional surfel map, as viewed by the robot operator during the debris task of the first DRC competition run. Color encodes height

clipped map—generated from robot's egocentric map described in Sect. 4—is transmitted over the communication link. The screen also shows the currently selected low-bandwidth image channel.

The left screen shows diagnostic information, including the ROS log (transmitted in the bursts), actuator temperatures, torques, and errors, and process status for nodes running on the robot and the operator station. The process monitoring is handled by the *rosmon* software.[7] The right screen shows a 2D height map of the environment and allows configuration of all system modules through a hierarchical GUI.

The support operators use notebooks connected to the operator station over Ethernet. Using the flexibility of ROS visualization tools, the notebooks offer views customized for the individual operator task.

[7] https://github.com/xqms/rosmon.

7.2 Motion Design

To support fast and flexible creation of motions by a human designer, we developed a set of motion editing tools. Motions are initially specified using a keyframe editor. At runtime, motions can be loaded, modified to fit the current situation, and finally executed by a player component.

The keyframe editor (see Fig. 16) is based on the standard ROS RViz graphical user interface. It shows the current robot state and the keyframe goal configuration as 3D models. Since the robot has a large number of independent endeffectors and internal joints, keyframes consist of multiple joint group states. For each joint group (e.g., the right arm), the user can specify either a target configuration in joint space, or a target endeffector pose in Cartesian space. Interpolation between the keyframes is controlled by specifying velocity constraints. Furthermore, the user can also control the amount of torque allowed in the motor controllers. Finally, the user can attach so-called frame tags to the keyframe, which trigger custom behavior, such as the wheel rolling with the motion of the leg. The tagging method allows the keyframe system to stay isolated from highly robot-specific behavior.

The described motion design method can be used offline to pre-design fixed motions, but it can also be used online to teleoperate the robot. In this case, the operator designs single-keyframe motions consisting of one goal configuration, which are then executed by the robot. The 3D map visualization can be displayed in the keyframe editor, so that the operator can see the current and target state of the robot in the perceived environment.

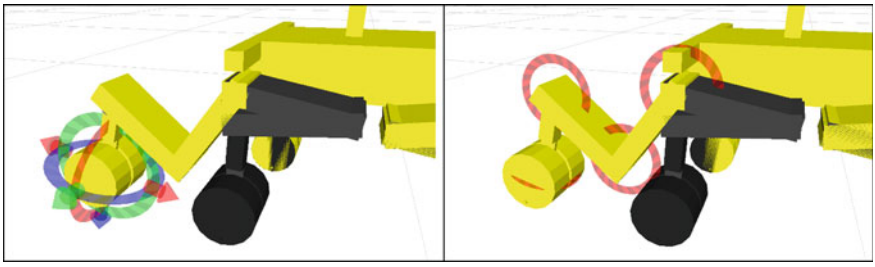

Fig. 16 Graphical user interface for keyframe editing. The user specifies a Cartesian target pose (left) or a target configuration in joint space (right). The yellow robot model displays the target configuration while the current robot configuration is shown in black

Fig. 17 Immersive telemanipulation. Left: Oculus Rift DK2 HMD. Center: Razer Hydra magnetic trackers. Right: Upper body operator using the HMD and trackers during DRC

7.3 Immersive Bimanual Telemanipulation

For intuitive and flexible manipulation, a designated upper body operator is responsible for controlling the robot, using two Razer Hydra[8] controllers (see Fig. 17). To give the operator an immersive feeling of being inside robot in its environment, he is wearing an Oculus Rift[9] which displays an egocentric view from the perspective of the robot which is based on the generated local multiresolution map. The Oculus Rift is an HMD which displays stereoscopic images and tracks the movement of the operator head in 6 DOF. It uses a combination of gyroscopes and acceleration sensors to estimate the rotation of the head and an additional camera-based tracking unit to determine the head position. The tracked head movements of the operator are used to update the stereoscopic view and allow the operator to freely look around in the current scene. In addition, transferred 2D camera images can be displayed in the view of the upper body operator to give him additional clues, as can be seen in the left part of Fig. 18. The selection and positioning of these views are performed by an additional support operator using a custom GUI (see Fig. 18).

The Razer Hydra hand-held controllers (see Fig. 17) use a weak magnetic field to sense the 6D position and orientation of the hands of the operator with an accuracy of 1 mm and 1°. The controllers have several buttons, an analog stick and a trigger. These controls map to different actions which the upper body operator can perform. The measured position and orientation of the operator hands are mapped to the position and orientation of the respective robot gripper to allow the operator to intuitively control them. We do not aim for a one-to-one mapping between the workspace of the robot and the reachable space of the magnetic trackers. Instead, differential commands are sent to the robot: The operator has to hold the trigger on the right or the left controller if he wants to control the respective arm. Vice versa, the operator needs to release the trigger to give up the control. This indexing technique enables the operator to move the robot grippers to the boundaries of the workspace in a comfortable way. Due to the limitation of the bandwidth, we send the desired 6D poses of the end-effectors with a limited rate of 5 Hz to the robot.

[8] https://en.wikipedia.org/wiki/Razer_Hydra.

[9] https://www3.oculus.com/en-us/rift/.

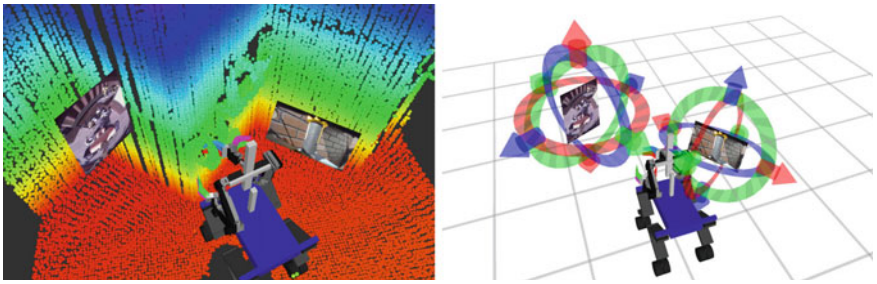

Fig. 18 Left: Third person view of the upper body operator display. Right: Same scene as seen by a support operator

For small-scale manipulation, the operator can switch to a precision mode. Here, motion is scaled down, such that large movements of the controllers result in smaller movements of the robot arms, thus enabling the operator to perform tasks with higher accuracy. The operator also has the ability to rotate the torso around the yaw axis using the analog stick on the left hand-held controller. The upper body operator can trigger basic torque-based open/close gripper motions with a button push. More complex grasps are configured by a support operator.

In addition, the upper body operator has the ability to move the point of view freely in the horizontal plane out of the egocentric view using the analog stick of the right Razer Hydra controller and can also flip the perspective by 180° at the push of a button. Both features allow the operator to inspect the current scene from another perspective.

The control system checks for self-collisions and displays the links which are nearly in collision color-coded to the operators. The system stops the execution of motion commands if the operator moves the robot further into nearly self-collision. We do not check collisions with the environment, as they are necessary to perform manipulation tasks.

7.4 Locomotion

During driving locomotion, the base velocity is controlled using a 4-axis joystick. The velocity components v_x, v_y, and ω are mapped to the three corresponding joystick axes, while the joystick throttle jointly scales all three components. The operator can control the footprint and base attitude using a custom base control GUI (see Fig. 19). The operator interface for semi-autonomous stepping (see Sect. 6.3) consists of a 2D height map (see Fig. 19) showing the robot footprint, COM, support polygon and candidate step locations.

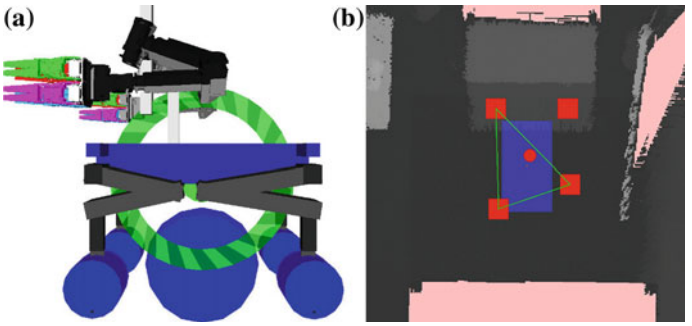

Fig. 19 Base control GUIs. **a** GUI for footprint and attitude control. The small blue wheels can be dragged with the mouse to adjust wheel positions. The blue sphere controls all wheels at once, and the green ring can be used to modify the pitch angle of the base. **b** 2D height map of the environment. The robot base is shown in blue, wheels are red rectangles, the COM is a red circle. The current support polygon is shown in green

7.5 Teleoperated Car Driving

For the DRC car task (see Sect. 8.1.1), we designed a custom operator interface (see Fig. 20) consisting of a special GUI and commercial gaming hardware controls (steering wheel and gas pedal). The steering wheel was mapped 1:1 to the rotation of the robot hand at the car steering wheel, while the pedal directly controlled the extension of the front right robot leg, which pressed down on the car gas pedal. During the driving, the responsible operator at the steering wheel uses high-resolution imagery (see Sect. 5.1.3) to keep track of the vehicle and the surrounding obstacles. While sitting in the car, the robot extends its right arm so that the operator is able to see the right front wheel and obstacles close to the car through the hand camera (see Fig. 20, image 4).

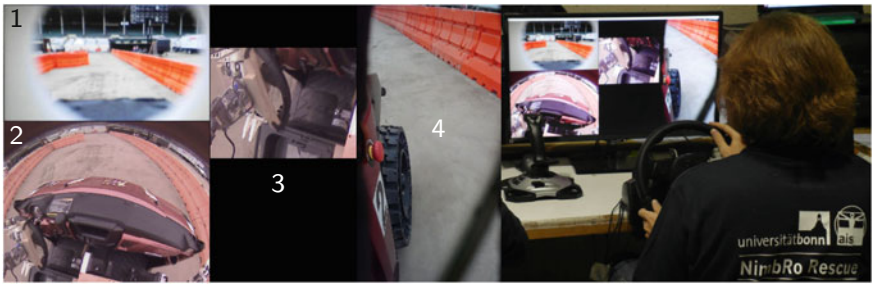

Fig. 20 User interface for the car driving task. Left: Camera view showing the center sensor head camera (1), the wide-angle overview camera (2), detail on the hand and gas pedal (3) and the right hand camera (4). Right: Operator using steering wheel and gas pedal during the driving task at DRC Finals

8 Evaluation at the DRC

The described system has been evaluated in several simulations and lab experiments as well as in the DARPA Robotics Challenge (DRC) Finals in June 2015.

8.1 DRC Finals

The DARPA Robotics Challenge consisted of eight tasks, three of which were mainly locomotion tasks, the other five being manipulation tasks. Additionally, the robot had to move from one task to the next. Since the overall time limit for all tasks was set at one hour, quick locomotion between the tasks was necessary for solving a large number of tasks. Please note that the Momaro robot design was targeted for more challenging and more numerous tasks, but DARPA lowered the number and the difficulty of the tasks shortly before the DRC Finals.

In general, the compliance in the legs not only provided passive terrain adaption, but also reduced the required model and kinematic precision for many tasks by allowing the robot trunk to move compliantly in response to environment contacts, e.g., while manipulating a valve. Furthermore, the strength of the leg actuators was also used for manipulation, for instance when opening the door by positioning the hand under the door handle and then raising the whole robot, thus turning the door handle upwards.

8.1.1 Car Driving and Egress

The car task featured a Polaris RANGER XP 900 vehicle (see Fig. 21), which the robot had to drive and exit. Since we did not have access to the car before the competition, we had only a few days at the Fairplex competition venue to determine how to fit the robot into the car and to design an appropriate egress motion. Even though the car task was the last we considered during the mechanical design, our base proved to be flexible enough to fit the robot in the car. We extended the gas pedal with a small lever to enable Momaro to push it with its front right leg. The steering wheel was fitted with two parallel wooden bars to enable Momaro to turn the wheel without grasping it by placing its fully opened gripper between the bars. Our driving operator only had few trial runs before the actual competition. In addition, the car engine could not be turned on during these trial runs, so the actual behavior of the car under engine power could not be tested and trained. Despite these limitations, we completed the car task successfully and efficiently on the preparation day and the two competition days. We conclude that our operator interface for driving (see Sect. 7.5) is intuitive enough to allow operation with very minimal training. In particular, the right hand image (see Fig. 20) was very helpful for keeping the appropriate distance to the obstacles.

Fig. 21 Momaro egresses from the car at the DARPA Robotics Challenge

While many teams opted to seat their robots dangerously close to the side of the car, so that they could exit with a single step, we placed the robot sideways on the passenger seat and used the robot wheels to slowly drive out of the car, stepping down onto the ground as soon as possible. Also, some teams made extensive modification to the car in order to ease the egressing progress, while we only added a small wooden foothold to the car to decrease the total height which had to be overcome in one step. We designed an egress motion consisting of both driving and stepping components, which made the robot climb backwards out of the passenger side of the vehicle. Momaro successfully exited the car on the trial day and in the first run of the competition (see Fig. 21). The attempt in the second run failed due to an operator mistake, resulting in an abort of the egress and subsequent reset of the robot in front of the door.

8.1.2 Door Opening

The first task to be completed after egressing from the vehicle is opening the door. In order to do this, our operators center the robot in front of the door, fold the delicate fingers out of the way, place the hand under the door handle, and use the robot legs to stand up and thus turn the door handle. Once inside the building, communication is degenerated.

Table 3 Manipulation results during the DRC Finals

Task	Success	Time (min:s)	
		1st run	2nd run
Door	2/2	2:25	0:27
Valve	2/2	3:13	3:27
Cutting	1/1	12:23	–
Switch	1/1	4:38	–
Plug	1/1	–	9:58

The listed times are calculated based on a recorded video feed. All attempted manipulation tasks were successfully solved. The listed times include the time for the locomotion from the previous task to the current task

In our first run, the first attempt at opening the door failed because the robot was positioned too far away from the door. After a small pose correction, the second attempt succeeded. The elapsed time for this task as well as all other attempted manipulation tasks are displayed in Table 3.

8.1.3 Turning a Valve

This task requires the robot to open a valve by rotating it counter-clockwise by 360°. The lower body operator positions the robot roughly in front of the valve. Then, a support operator marks the position and orientation of the valve for the robot using an 6D interactive marker (Gossow et al. 2011) in a 3D graphical user interface. After the valve is marked, a series of parameterized motion primitives, which use the marked position and orientation, are executed by the support operator to fulfill the task. First, the right hand is opened widely and the right arm moves the hand in front of the valve. The correct alignment of the hand and the valve is verified using the camera in the right hand and the position of the hand is corrected if the alignment is insufficient. Next, we perform the maximum clockwise rotation of the hand and the flexible finger tips close around the outer part of the valve to get a firm grasp of the valve. Due to kinematic constraints, we can only turn the hand by 286° (5 rad). After that, the hand opens again and the sequence is repeated until the valve is fully opened. We demonstrated turning the valve successfully in both runs (see Fig. 22). During the first run, one finger tip of the right gripper slipped into the valve and was slightly damaged when we retracted the end-effector from the valve. We continued the run without problems.

8.1.4 Cutting Drywall

The cutting task requires the robot to grasp one of four supplied drill tools and use the tool to remove a marked circle from a piece of drywall by cutting around it. We chose to use a tool with a slide switch that only needs to be triggered once. One finger of

Fig. 22 Left: Inserting the plug as seen from the right hand camera. Middle: Momaro turns the valve. Right: Flipping the switch as seen from the top-down camera

the right hand was equipped with a small bump to actuate the switch. Since the tool switches off automatically after five minutes, we did not need to design a switch-off mechanism. The tool is grasped by the upper body operator using the Razer Hydra controller by aligning the gripper to the tool and triggering a predefined grasp. The arm is then retracted by the upper body operator and a support operator triggers a motion primitive which rotates the hand by 180°. As the first grasp does not close the hand completely, the tool can now slip into the desired position. A support operator executes another motion to close the hand completely and switch the tool on. After tool activation is confirmed by rising sound volume from the right hand microphone, the upper body operator positions the tool in front of the drywall. We fit a plane into the wall in front of the robot to automatically correct the angular alignment. The lower body operator then drives the base forward, monitoring the distance to the wall measured by the infrared distance sensor in the right hand. A parameterized motion primitive is then used to cut a circular hole into the drywall. When the task is completed, the tool is placed on the floor.

In our first run, we grasped the tool successfully and rotated it upside down (see Fig. 23). Some manual adaptation of the gripper in joint space was necessary since the tool was initially not grasped as desired. As we tried to cut the drywall, we became aware that the cutting tool was not assembled correctly. Therefore, our first run was paused by the DARPA officials and the cutting tool was replaced. The lost time was credited to us. During our second cutting attempt, our parameterized cutting motion primitive was not executed correctly as the robot was not properly aligned to the drywall. Consequently, the automated cutting motion did not remove all designated material. We noticed this error during the execution of the motion and a support operator moved the right arm manually upwards, breaking the drywall and fulfilling the task.

8.1.5 Operating a Switch

This task was the surprise task for the first run. The task is to flip a big switch from its on-position into its off-position. After the robot was driven in front of the switch, the upper body operator solves this task on his own. He closes the fingers of the right hand half way using a predefined motion and then moves the hand towards the

Fig. 23 Top Left: Grasping the cutting tool as seen from the right hand camera. Right: Same scene as seen from the top-down camera. Middle Left: Grasp used to switch on the tool. Bottom Left: Momaro cutting the drywall as seen from a sensor head camera

lever of the switch. As soon as the hand encloses the lever, the robot base is used to lower the whole robot, thus pushing the lever down. Since we did not have a mockup of the switch, we were not able to train this task prior to the run. Nevertheless, we succeeded in our first attempt.

8.1.6 Plug Task

This task was the surprise task for the second run. The task was to pull a plug from a socket and insert it into a different socket which was located 0.5 m horizontally away from the first socket. For this task, we added additional distal finger segments to the left hand of the robot to increase the surface area which has contact with the plug. During this task, a support operator controls the left gripper using a 6D interactive marker. The interactive marker allows to move the gripper exclusively in a fixed direction, which is difficult using the hand-held controllers. During the run, it took us several attempts to solve the plug task. We used the camera in the right hand to verify that we successfully inserted the plug into the socket as can be seen in Fig. 22.

8.1.7 Traversing Debris

Most teams with a legged robot chose to walk over the special terrain field. Instead, we chose to drive through the debris field using the powerful wheels. During the trial run and first competition run, the robot simply pushed through the loose obstacles and drove over smaller ones quite fast (see Fig. 24). To maximize stability, we kept the COM very low by completely folding the legs. During the second competition run, Momaro unfortunately got stuck in a traverse that was part of the debris. After a longer recovery procedure, the robot still managed to solve the task, although several actuators failed due to overheating.

8.1.8 Stairs

Sadly, we could not demonstrate the stairs task during the DRC Finals due to development time constraints and the debris entanglement in the second run. However, we were able to show that the robot is capable of climbing stairs directly afterwards in an experiment in our lab (see Fig. 25). To do so, the robot also leverages its base as a ground contact point, increasing stability of the motion and allowing to use both forelegs simultaneously to lift the base onto the next step.[10] The execution time for the experiment was only 149 s.

8.1.9 DRC Summary

Our team NimbRo Rescue solved seven of the eight DRC tasks and achieved the lowest overall time (34 min) under all DRC teams with seven points[11]— the next team took 48 min. This good overall result demonstrated the usefulness of the hybrid locomotion design, the flexibility of our approach, and its robustness in the presence of unforeseen difficulties.

As with any teleoperated system, operator training is an important aspect of preparation, if not the most important one. Due to the tight time schedule, our team started testing entire runs (omitting driving, egressing and the stair task as mentioned before) regularly about 2 weeks before the DRC finals. Before that time, only smaller manipulation tests were possible, since the robot hardware was not finished yet. We feel that this short training phase was only possible due to the large number of operators, who could independently train on their particular interface and improve it. The specialization of operators was also seen with concern, since any operator being unavailable for any reason, e.g., due to sickness, would severely limit the crew's abilities. For this reason, we also trained backup operators for the central tasks, in particular telemanipulation with the Oculus/Razer setup, locomotion control, and system monitoring. Fortunately, all operators were able to participate in the DRC Finals.

[10] Video: https://www.youtube.com/watch?v=WzQDBRjHRH8.

[11] Video: https://www.youtube.com/watch?v=NJHSFelPsGc.

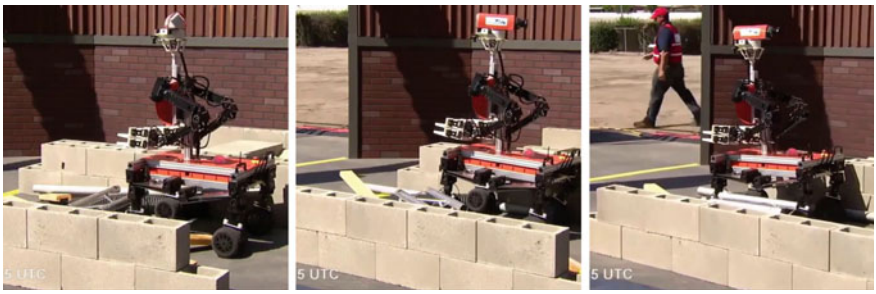

Fig. 24 Momaro pushes through loose debris at the DARPA Robotics Challenge

Fig. 25 Momaro climbs stairs, using a specially designed stair gait

We estimated detailed task and locomotion durations from the available video footage for the top five teams (see Table 4). It is clear that these numbers are quite noisy as many factors influence the execution time needed in the particular run. Nevertheless, some trends can be observed. The winning team KAIST was fastest in five tasks, but took longest for locomotion, because of transitions from standing to the kneeling configuration. Our team NimbRo Rescue was fastest for the egress task and needed the shortest time for locomotion, due to the fast and flexible omnidirectional driving of our robot Momaro.

8.2 Evaluation of Bimanual Telemanipulation

During the DRC Finals, we rarely used more than one end-effector at a time. One example of using both hands is the plug task, where we used the right end-effector camera to observe the motions of the left gripper. To evaluate the bimanual teleoperation capabilities of our system, we designed an additional task, which exceeds the requirements of the DRC Finals.

The task is to connect two flexible unmodified water hoses (see Fig. 26). No locomotion is needed during this task, as the hoses are placed within the reachable workspace of the robot arms with both hoses supported in graspable height. Since the hoses are flexible, the operator has to grasp both connectors with the left and right gripper, respectively, in order to push them together.

Table 4 Task and locomotion timings for the top five DRC teams

Team	Day	DRC Tasks								Locomotion
		Car	Egress	Door	Valve	Wall	Rubble	Surprise[a]	Stairs	
KAIST	2	**50**	253	**70**	**42**	**612**	71	435	**275**	856
IHMC	2	90	293	142	197	624	240	342	288	793
Tartan R.	1	117	836	524	55	813	111	**72**	298	491
NimbRo R.	1	120	**219**	103	151	742	110	192	–	**477**
Robosimian	1	266	518	126	86	798	**54**	311	–	713

The better of the two runs is shown. The times have been roughly estimated from captured videos at DRC Finals and are by no means official. The "Locomotion" column shows the total time spent driving/walking between the tasks. Fine-alignment in front of a task is included. Locomotion during a task is not counted

[a]The surprise task was different between the two runs and is thus not comparable

One support operator assisted the trained upper body operator during the task by controlling the camera images which are displayed in his HMD and by triggering grasps. A monoscopic view from the perspective of the upper body operator can be seen in the right part of Fig. 26. The hoses as well as the support traverses are clearly visible in the 3D point cloud, which provides the operator a good awareness of the current situation. 2D camera images are displayed to aid the operator with additional visual clues. The operators were in a different room than the robot during the experiments and received information over the state of the robot and its environment only from the robot sensors. The communication bandwidth was not limited.

We performed the hose task 11 times in our lab. The execution of one trial was stopped, as the upper body operator moved the right arm into the base of the robot as he was grasping for the right hose. The results of the remaining 10 executions of the experiments are shown in Table 5. The task consists of three parts which are separately listed: 1. Grab the left hose with the left gripper, 2. Grab the right hose with the right gripper, and 3. Connect both hoses. On average, slightly more than three minutes were needed to complete the whole task. The hardest part of the task was to establish the actual connection between both hoses, which accounted on average for more than half of the total elapsed time.

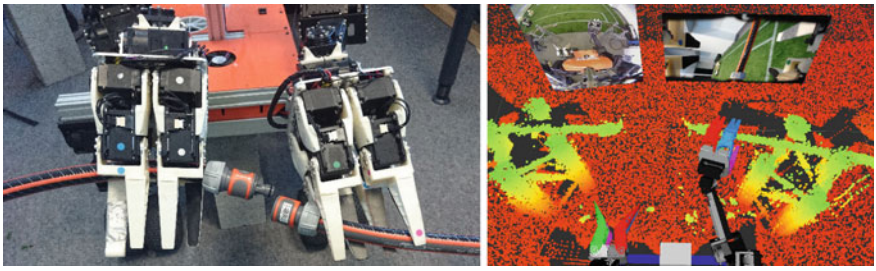

Fig. 26 Connecting hoses. Left: Momaro connecting two hoses. Right: Upper body operator view during the hose task

Table 5 Execution times for the hose task (10 trials)

Task	Time (min:s)				
	Average	Median	Minimum	Maximum	Std. dev.
Left grasp	0:44	0:38	0:27	1:20	0:16
Right grasp	0:45	0:40	0:34	1:04	0:10
Connect	1:36	1:32	1:07	2:04	0:21
Total	3:04	2:57	2:21	3:51	0:28

9 Lessons Learned from the DRC

Our participation in the DRC was extremely valuable for identifying weak points in the described system which resulted in task or system failures. It also demonstrated which design choices led to favorable performance of the system.

9.1 Mechatronic Design

Using relatively low-cost, off-the-shelf actuators has drawbacks. In particular, the actuators overheat easily under high torque, for example if the robot stays too long in strenuous configurations. Such configurations include standing on the stairs with one leg lifted. In teleoperated scenarios, delays can occur and are not easily avoided. This hardware limitation prevented us from attempting the stairs task in our first run, where we would have had ample time (26 min) to move the robot up the stairs with teleoperated motion commands. Any delay in reaching intermediate stable configurations would have resulted in overheating and falling. It seems that many other teams put considerable effort in active cooling of the actuators, which would reduce this problem. As a consequence, an improved Momaro design would include cooled (or otherwise stronger) actuators, especially in the legs. Also, future work will focus on further exploiting the advantages of the design by investigating autonomous planning of hybrid driving and stepping actions, thus allowing fluid autonomous locomotion over rough terrain and avoiding overheating of the actuators.

Designing the legs as springs allowed us to ignore smaller obstacles and also provides some compliance during manipulation, reducing the needed precision. However, we also encountered problems: During our first competition run, our field crew was worried that the robot would fall during the drill task, because one leg had moved unintentionally far below the base, reducing the support polygon size. The deviation was entirely caused by the springs and thus not measurable using joint encoders. Future compliant designs will include means to measure deflection of the compliant parts, such that autonomous behaviors and the operator crew can react to unintended configurations.

9.2 Operator Interfaces

In particular, operator mistakes caused failures for many teams as the reports collected in (DRC-Teams 2015) indicate. Our second run suffered from an operator mistake (triggering the wrong motion) as well, which could have been avoided by a better user interface. In particular, our operator interfaces were not designed to protect against dangerous operator commands. In the future, we will strive to anticipate possible operator mistakes and develop means to prevent them. Also, unanticipated

situations could be detected on the robot side, resulting in an automatic pause of the semi-autonomous behaviors. More time for operator training would also reduce the number of operator mistakes.

As a second issue in our second run, the robot got entangled with a piece of debris. Since we did not train this situation, our operator crew lost a lot of time and power to get out of it. As a consequence, the robot actuators overheated, which made the situation even worse. The only possible conclusion for us is that our situational awareness was not good enough to avoid the situation, which might be improved by mounting additional sensors in front of the robot. Additionally, recovery from stuck/overheated situations should be assisted by the user interface and trained by the operator crew.

9.3 Sensors

It may be not a new insight, but especially our group—mainly focusing on autonomy—was surprised by the usefulness of camera images for teleoperation. In the (autonomous) robotics community there seems to be a focus on 3D perception, which is understandable for autonomous operation. But color cameras have distinct advantages over 3D sensors: They are cheap, work in harsh light conditions, and images are easily interpretable by humans. As a result, Momaro carries seven cameras, placed in strategic positions to be able to correctly judge situations from remote. This strongly augments the 3D map from laser measurements.

9.4 Preparation Time

Our team found a viable solution for the stairs task in the night before our second run. One could argue, that we would have needed one more day of preparation to solve all tasks. Of course, other factors could have easily kept us from reaching this goal as well, as seen in our second run. Nevertheless, while preparation time for competitions is always too short, in our case it was maybe especially so.

10 Towards Autonomous Operator Assistance

During the DARPA Robotics Challenge, little autonomy was required. The limited number of specific tasks meant that our large operator crew could train sufficiently to become fast and experienced at solving the challenge using teleoperation. However, in order to succeed at real-world rescue operations, the system (including the operators) needs to be much more flexible. We argue that too much reliance on teleoperation requires constant focus of the operators on low-level tasks, which could

be delegated to autonomous behaviors, freeing the operators to devote more of their time on high-level decisions such as where to explore next, and how to best apply the capabilities of the robotic system to solve a particular problem. In this section, we present our initial steps towards this goal in the framework of the CENTAURO project,[12] a European H2020 research and innovation action aiming to develop a human-robot symbiotic system for disaster response with focus on both immersive teleoperation and autonomous operator assistance functions.

10.1 Allocentric Mapping

During the DRC, we incorporated IMU measurements, wheel odometry measurements and registration results from the local mapping to estimate the motion of the robot. While this information allows us to control the robot from the operator station and to track its pose over a short period, it drifts over time. When the robot is teleoperated, this drift is negligible. For autonomous operation in larger environments, however, it can quickly become problematic.

To overcome drift and to localize the robot with respect to a fixed frame, we build an allocentric map from local multiresolution maps acquired at different view poses (Droeschel et al. 2017). The local maps are used to construct a pose graph, where every node corresponds to a view pose. Nodes are connected by spatial constraints derived from aligning local multiresolution maps, describing the relative pose between them. To track the current pose of the robot in the allocentric frame, the current local map is registered towards the closest node in the graph. By aligning the dense local map—instead of the relative sparse 3D scan—to the pose graph, we gain robustness, since information from previous 3D scans is incorporated. The graph is extended with the current view pose, if the robot moved sufficiently far. Furthermore, we connect close-by view poses to detect loop closures, allowing to minimize drift if the robot reenters known parts of the environment.

When the pose graph is extended or a loop closure has been detected, we optimize the trajectory estimate given all relative pose observations using the g^2o framework (Kuemmerle et al. 2011). This optimization yields maximum likelihood estimates of the view poses, and results in a globally consistent allocentric map of the environment.

Figure 27 shows the resulting allocentric map generated from the dataset of our first competition run. In addition to the allocentric map, selected local multiresolution maps of the pose graph are shown. Although reference data is not available, one can see that the resulting allocentric map is globally consistent and accurate, as indicated by the straight walls and plain floor. Also the local maps look clear and accurate.

[12] http://www.centauro-project.eu.

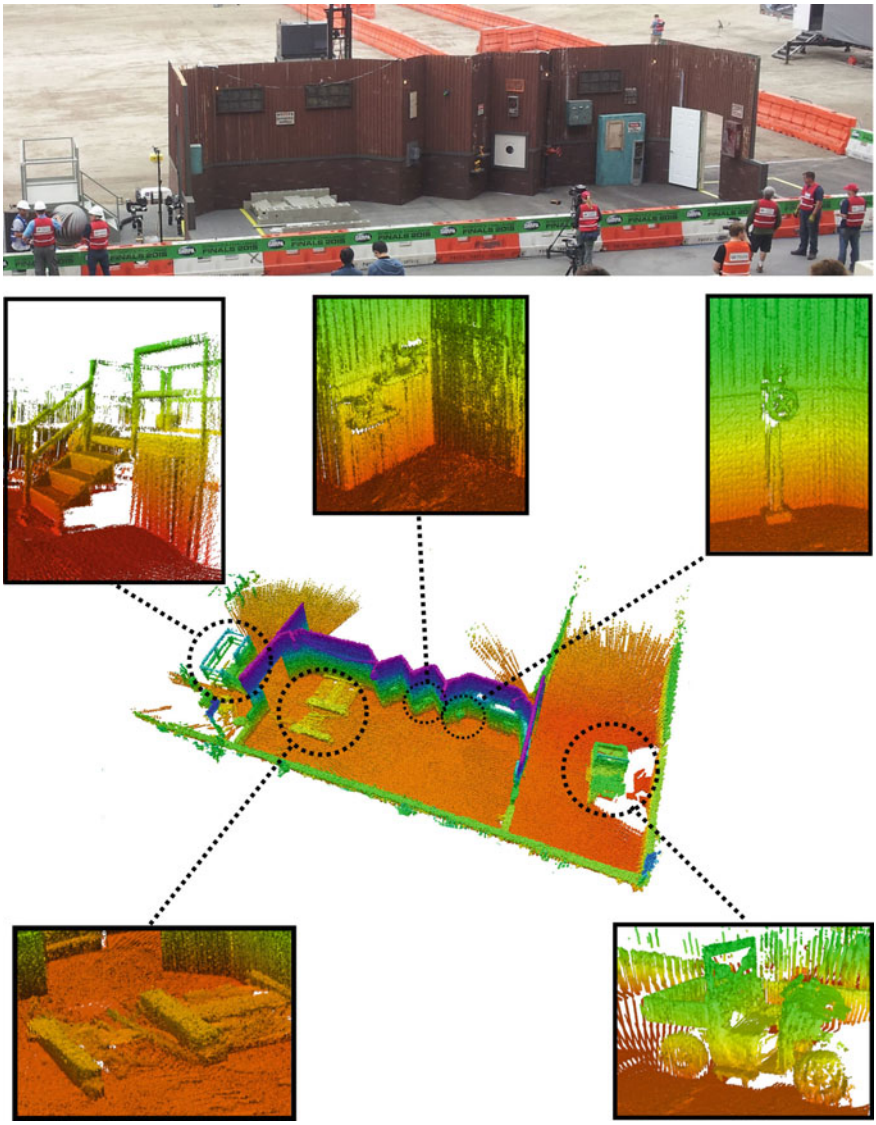

Fig. 27 Top: The mock-up disaster scenario of the DRC. Bottom: The resulting allocentric map generated from the data of our first competition run. Color encodes the height from ground

10.2 Arm Trajectory Optimization

During the DRC challenge, one of our operators was responsible for directly controlling both arms using magnetic trackers (see Sect. 7.3). Since our ultimate goal is to free the operators from such direct control tasks, we investigated methods for motion planning after the challenge.

Pavlichenko and Behnke (2017) describe a method for trajectory optimization under multiple criteria, such as collision avoidance, orientation, and torque constraints. The planner is a variant of the well-known STOMP algorithm for trajectory optimization with the trajectory optimization being performed in two stages for efficiency. Figure 28 shows an experiment on Momaro incorporating the 7 DOF of the left robot arm and its trunk yaw joint.

10.3 Autonomous Hybrid Driving-Stepping Locomotion

We also investigated autonomous locomotion planning capabilities for Momaro. While the challenge itself did not require stepping motions except for the stereotyped car egress and stair tasks, the general ability to overcome non-drivable height differences is very important in rescue scenarios. Manually specifying stepping actions is quite tedious, however, and requires constant operator attention. A robust autonomous navigation planner capable of both driving and stepping actions is thus required.

As described by Klamt and Behnke (2017), the height map representation from Sect. 4.3 is used to plan full motion sequences consisting of driving motions, weight shifts, and stepping motions. Figure 29 shows a very challenging simulated scenario designed to demonstrate the capabilities of the planner, requiring all types of possible

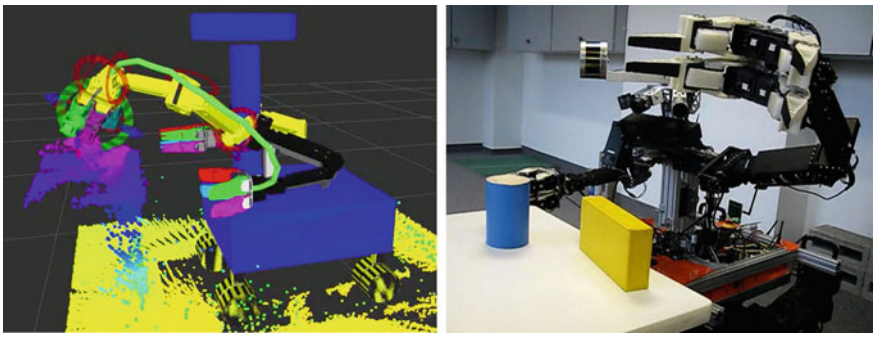

Fig. 28 Trajectory optimization experiment on Momaro. The goal is to reach the blue cup, which is obstructed behind the yellow box. Left: Planned trajectory. Right: Momaro executing the planned trajectory

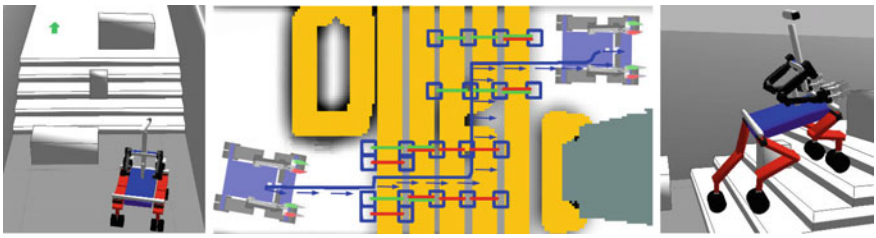

Fig. 29 Autonomous hybrid locomotion in simulation. This scenario was designed to show the capabilities of the planner. Note that this scenario is difficult even for tracked vehicles. Left: Initial situation with robot in front of the obstacle. Center: Generated plan with COM trajectory and generated stepping motions for front and rear legs in red and green, respectively. Right: Plan execution

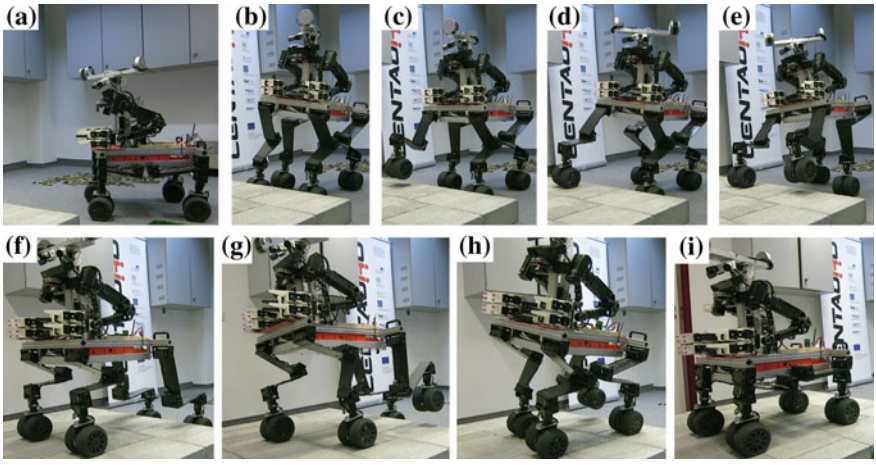

Fig. 30 Autonomous hybrid locomotion with the real robot: Driving to and stepping on an elevated platform. **a** Driving to the step. **b–d** First step. **e–h** Second, third, and fourth step. **i** Goal pose on the platform

motions, including omnidirectional driving. In addition to simulation experiments, we also tested the planner on the real robot (see Fig. 30). For details on the planning framework, we refer to Klamt and Behnke (2017).

10.4 DLR SpaceBot Camp 2015

Our team also participated with the Momaro system in the DLR SpaceBot Camp in November 2015 (Kaupisch et al. 2015). In contrast to the DRC, this challenge required mostly autonomous operation, which was enforced by high communication latency and only small windows where new commands could be uploaded.

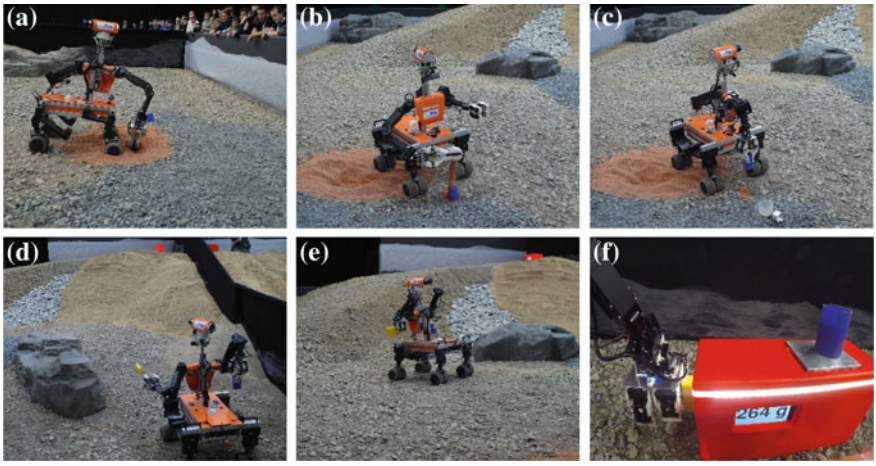

Fig. 31 Momaro solves all tasks of the DLR SpaceBot Camp 2015. **a** Scooping up a soil sample. **b** Filling the soil sample into the cup object. **c** Discarding the scoop and grasping the cup. **d** Locating and grasping the yellow battery object. **e** Driving to the base station object. **f** Assembly task at the base station

To address this situation, we developed a high-level control framework for supervised autonomy, which allowed the operator to specify and update missions consisting of waypoints with associated manipulation actions and supervise the execution. Our capable teleoperation mechanisms developed during the DRC served as a backup, should the autonomy fail. Our system was the only system to complete all tasks including the optional soil sample task (see Fig. 31). For more information, we refer to Schwarz et al. (2016).

11 Conclusion

In this paper, we presented the mobile manipulation robot Momaro and its operator station and evaluated its performance in the DARPA Robotics Challenge, the DLR SpaceBot Camp 2015, and several lab experiments. Novelties include a hybrid mobile base combining wheeled and legged locomotion and our immersive approach to intuitive bimanual manipulation under constrained communication. The great success of the developed robotic platform and telemanipulation interfaces at the DRC has demonstrated the feasibility, flexibility and usefulness of the design.

To solve complex manipulation tasks, our operators currently rely on 3D point clouds, visual and auditory feedback, and joint sensors from the robot. Additional touch and force-torque sensing in combination with a force feedback system for the upper body operator could potentially improve the manipulation capabilities of the human-robot system. This could, for example, be beneficial for peg-in-hole tasks

such as the plug task during the DRC or the hose task, which require precise and dexterous manipulation skills.

Our telemanipulation system has currently only a low degree of autonomy and instead requires multiple human operators to control it. This allows our team to easily react to unforeseen events. However, the number of operators needed is quite high and so many trained operators are not always available. We progressed towards more autonomous functions for assisting the operators, but full integration of such a system is yet to be demonstrated.

Acknowledgements This work was supported by the European Union's Horizon 2020 Programme under Grant Agreement 644839 (CENTAURO) and by Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR) under Grant No. SORA1413.

References

Adachi, H., Koyachi, N., Arai, T., Shimiza, A., & Nogami, Y. (1999). Mechanism and control of a leg-wheel hybrid mobile robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Vol. 3, pp. 1792–1797).

Ballantyne, G. H., & Moll, F. (2003). The da Vinci telerobotic surgical system: The virtual operative field and telepresence surgery. *Surgical Clinics of North America*, *83*(6), 1293–1304.

Borst, C., Wimbock, T., Schmidt, F., Fuchs, M., Brunner, B., Zacharias, F., et al. (2009). Rollin' Justin—Mobile platform with variable base. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1597–1598).

Buss, S. R., & Kim, J.-S. (2005). Selectively damped least squares for inverse kinematics. *Graphics, GPU, and Game Tools*, *10*(3), 37–49.

Cho, B.-K., Kim, J.-H., & Oh, J.-H. (2011). Online balance controllers for a hopping and running humanoid robot. *Advanced Robotics*, *25*(9–10), 1209–1225.

Cigolle, Z. H., Donow, S., & Evangelakos, D. (2014). A survey of efficient representations for independent unit vectors. *Journal of Computer Graphics Techniques*, *3*(2).

DRC-Teams. (2015). What happened at the DARPA Robotics Challenge? www.cs.cmu.edu/~cga/drc/events.

Droeschel, D., Schwarz, M., & Behnke, S. (2017). Continuous mapping and localization for autonomous navigation in rough terrain using a 3d laser scanner. *Robotics and Autonomous Systems*, *88*, 104–115.

Droeschel, D., Stückler, J., & Behnke, S. (2014). Local multi-resolution representation for 6d motion estimation and mapping with a continuously rotating 3d laser scanner. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5221–5226).

Endo, G., & Hirose, S. (2000). Study on roller-walker (multi-mode steering control and self-contained locomotion). In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Vol. 3, 2808–2814).

Gossow, D., Leeper, A., Hershberger, D., & Ciocarlie, M. (2011). Interactive markers: 3-d user interfaces for ROS applications. *IEEE Robotics & Automation Magazine*, *4*(18), 14–15.

Hagn, U., Konietschke, R., Tobergte, A., Nickl, M., Jörg, S., Kübler, B., et al. (2010). DLR MiroSurge: A versatile system for research in endoscopic telesurgery. *International Journal of Computer Assisted Radiology and Surgery (IJCARS)*, *5*(2), 183–193.

Halme, A., Leppänen, I., Suomela, J., Ylönen, S., & Kettunen, I. (2003). WorkPartner: Interactive human-like service robot for outdoor applications. *International Journal of Robotics Research (IJRR)*, *22*(7–8), 627–640.

Hebert, P., Bajracharya, M., Ma, J., Hudson, N., Aydemir, A., Reid, J., et al. (2015). Mobile manipulation and mobility as manipulation-design and algorithms of robosimian. *Journal of Field Robotics (JFR)*, *32*(2), 255–274.

Huang, T., Yang, G., & Tang, G. (1979). A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *27*(1), 13–18.

Johnson, M., Shrewsbury, B., Bertrand, S., Wu, T., Duran, D., Floyd, M., et al. (2015). Team IHMC's lessons learned from the DARPA robotics challenge trials. *Journal of Field Robotics (JFR)*, *32*(2), 192–208.

Kaupisch, T., Noelke, D., & Arghir, A. (2015). DLR spacebot cup—Germany's space robotics competition. In *Proceedings of the Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*.

Kim, M.-S., & Oh, J.-H. (2010). Posture control of a humanoid robot with a compliant ankle joint. *International Journal of Humanoid Robotics*, *07*(01), 5–29.

Klamt, T. & Behnke, S. (2017). Anytime hybrid driving-stepping locomotion planning. In *Accepted for International Conference on Intelligent Robots and Systems (IROS)*.

Kot, T. & Novák, P. (2014). Utilization of the Oculus Rift HMD in mobile robot teleoperation. In *Applied Mechanics and Materials* (Vol. 555, pp. 199–208). Trans Tech Publications.

Kröger, T. (2011). Opening the door to new sensor-based robot applications—The Reflexxes Motion Libraries. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Kron, A., Schmidt, G., Petzold, B., Zäh, M., Hinterseer, P., Steinbach, E., et al. (2004). Disposal of explosive ordnances by use of a bimanual haptic telepresence system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1968–1973).

Kuemmerle, R., Grisetti, G., Strasdat, H., Konolige, K., & Burgard, W. (2011). G2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Liegeois, A. (1977). Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, *7*(12), 868–871.

Lim, J. & Oh, J.-H. (2015). Backward ladder climbing locomotion of humanoid robot with gain overriding method on position control. *Journal of Field Robotics (JFR)*.

Martins, H., & Ventura, R. (2009). Immersive 3-D teleoperation of a search and rescue robot using a head-mounted display. In *Proceedings of the international Conference on Emerging Technologies and Factory Automation (ETFA)*.

Mehling, J., Strawser, P., Bridgwater, L., Verdeyen, W., & Rovekamp, R. (2007). Centaur: NASA's mobile humanoid designed for field work. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2928–2933).

Pavlichenko, D. & Behnke, S. (2017). Efficient stochastic multicriteria arm trajectory optimization. In *Accepted for International Conference on Intelligent Robots and Systems (IROS)*.

Raibert, M., Blankespoor, K., Nelson, G., Playter, R., et al. (2008). BigDog, the rough–terrain quadruped robot. In *Proceedings of the 17th World Congress, The International Federation of Automatic Control* (pp. 10823–10825), Seoul, Korea

Roennau, A., Kerscher, T., & Dillmann, R. (2010). Design and kinematics of a biologically-inspired leg for a six-legged walking machine. In *3rd IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)* (pp. 626 –631).

Satzinger, B., Lau, C., Byl, M., & Byl, K. (2014). Experimental results for dexterous quadruped locomotion planning with RoboSimian. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*.

Schwarz, M., Beul, M., Droeschel, D., Schüller, S., Periyasamy, A. S., Lenz, C., et al. (2016a). Supervised autonomy for exploration and mobile manipulation in rough terrain with a centaur-like robot. *Frontiers in Robotics and AI*, *3*, 57.

Schwarz, M., Rodehutskors, T., Schreiber, M., & Behnke, S. (2016b). Hybrid driving-stepping locomotion with the wheeled-legged robot momaro. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Semini, C., Tsagarakis, N., Guglielmino, E., Focchi, M., Cannella, F., & Caldwell, D. (2011). Design of HyQ—A hydraulically and electrically actuated quadruped robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *225*(6), 831–849.

Smith, C., Christensen, H., et al. (2009). Wiimote robot control using human motion models. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 5509–5515).

Stentz, A., Herman, H., Kelly, A., Meyhofer, E., Haynes, G. C., Stager, D., et al. (2015). CHIMP, the CMU highly intelligent mobile platform. *Journal of Field Robotics (JFR)*, *32*(2), 209–228.

Stückler, J., Droeschel, D., Gräve, K., Holz, D., Schreiber, M., Topalidou-Kyniazopoulou, A., et al. (2014). Increasing flexibility of mobile manipulation and intuitive human-robot interaction in RoboCup@Home. In *RoboCup 2013: Robot World Cup XVII* (pp. 135–146). Springer.

Stückler, J., Schwarz, M., Schadler, M., Topalidou-Kyniazopoulou, A., & Behnke, S. (2015). NimbRo Explorer: Semiautonomous exploration and mobile manipulation in rough terrain. *Journal of Field Robotics (JFR)*.