Safeguarding Autonomy Through Intelligent Shared Control

Jared Giesbrecht\textsuperscript{a}, Ho-Kong Ng\textsuperscript{b}, Mingfeng Zhang\textsuperscript{b}, John Tang\textsuperscript{b}, Michel Bondy\textsuperscript{b}, Piotr Jasiobedzki\textsuperscript{b}

\textsuperscript{a}Defence Research & Development Canada
Box 4000 Station Main
Medicine Hat, AB, CA
T1A 8K6

\textsuperscript{b}MacDonald, Dettwiler and Associates Inc. (MDA)
9445 Airport Road, Brampton, ON, CA
L6S 4J3

PROCEEDINGS OF SPIE

Date of Publication from Ext Publisher: May 2017

Terms of Release: This document is approved for Public release.
IMPORTANT INFORMATIVE STATEMENTS

Disclaimer: This document is not published by the Editorial Office of Defence Research and Development Canada, an agency of the Department of National Defence of Canada, but is to be catalogued in the Canadian Defence Information System (CANDIS), the national repository for Defence S&T documents. Her Majesty the Queen in Right of Canada (Department of National Defence) makes no representations or warranties, express or implied, of any kind whatsoever, and assumes no liability for the accuracy, reliability, completeness, currency or usefulness of any information, product, process or material included in this document. Nothing in this document should be interpreted as an endorsement for the specific use of any tool, technique or process examined in it. Any reliance on, or use of, any information, product, process or material included in this document is at the sole risk of the person so using it or relying on it. Canada does not assume any liability in respect of any damages or losses arising out of or in connection with the use of, or reliance on, any information, product, process or material included in this document.

This document was reviewed for Controlled Goods by Defence Research and Development Canada (DRDC) using the Schedule to the Defence Production Act.
Safeguarding autonomy through intelligent shared control

Jared Giesbrecht
Ho-Kong Ng
Mingfeng Zhang
John Tang
Michel Bondy
Piotr Jasiobedzki
Safeguarding Autonomy Through Intelligent Shared Control

Jared Giesbrechta, Ho-Kong Ngb, Mingfeng Zhangb, John Tangb, Michel Bondyb, Piotr Jasiobedzkib
aDefence Research & Development Canada, Box 4000 Station Main, Medicine Hat, AB, CA, T1A 8K6; bMacDonald, Dettwiler and Associates Inc. (MDA), 9445 Airport Road, Brampton, ON, CA, L6S 4J3

ABSTRACT

This paper overviews the development and operator testing of a shared autonomy system for small unmanned ground vehicles operating in indoor environments. The project focused on creating driving assistance technologies to reduce the burden of performing low-level tasks when operating in cluttered or difficult areas by sharing control between the operator and the autonomous software. The system also provides a safety layer to prevent the robot from becoming disabled due to operator error or environmental hazards. Examples of developed behaviours include obstacle proximity warning, centering the vehicle through narrow doorways, wall following during long traversals, tip-over indicator, stair climbing aid, and retreat from communications loss. The hardware and software were integrated on a QinetiQ Talon IV robot and tested by military operators in a relevant environment.

Keywords: Shared control, semi-autonomous, obstacle avoidance, stair climbing, robot control

1. INTRODUCTION

1.1 Limits of UGV Tele-Operation

Tele-operation of Unmanned Ground Vehicles (UGVs) can be a difficult task in cluttered or poorly structured environments, with several sources documenting performance issues. A typical mobile robot system has limited physical and video sensing, feedback delays limiting reaction time, and unreliable communication, particularly in indoor or urban environments. Furthermore, there are significant mobility challenges for UGV platforms that are small enough to be man-portable and operate indoors, but not large enough for many of the obstacles encountered. However, the primary problem is a lack of operator situational awareness of the mobile robot’s condition and environment.

Sensors on most UGVs have a number of constraints which limit the information that can be passed to the operator. For example, for a video camera on a UGV it would be ideal to have a complete 360º field of view to perceive the immediate environment, while at the same time having infinite resolution to see details required for manipulation tasks or long-range route planning. Unfortunately, aside from cost restrictions, there is limited screen space on the operator control station, limited sensor space on the robot, and limited radio bandwidth to transmit data. Several attempts have been made to improve operator awareness through the use of virtual reality and other more immersive control stations.

A second factor inhibiting situational awareness for UGV operators is the lack of proprioceptive feedback to the operator. For a human walking through a cluttered environment, he has a sense where all his limbs are, and can feel if he is slipping, tipping over, or has bumped into an obstacle. However, through an operator control station these conditions are not so obvious. Various projects have developed systems to provide this information to the operator through haptic feedback, auditory signals, etc. However, this approach is again limited by sensing available onboard the robot and radio bandwidth. Furthermore, this approach leads to expensive and complicated robots and control stations, and may further contribute to overwhelming the operator.

As such, it is reasonable to install some degree of autonomy onboard the UGV to accommodate these limitations and to reduce the operator’s mental burden, allowing him to focus on mission objectives. However, the current state-of-the-art does not allow for completely autonomous operations, particularly for poorly defined missions or in unknown and complex environments. The result is that human control and interaction will be necessary for the foreseeable future. Unfortunately, because of the difficulties in perception indicated, this will continue to limit the speed at which UGV missions can be conducted.
The goal then is to seamlessly integrate autonomy where it can improve robot safety, reduce operator burden, and increase execution speed while refraining from interfering with operator intentions. This work attempts to address some of the difficulties with current tele-operated systems through the implementation of practical, shared control semi-autonomous behaviours using cost-effective hardware and software that can be retro-fitted to existing UGV systems.

1.2 Shared Semi-Autonomous Control

The concepts of using shared control, semi-autonomous behaviours, and guarded tele-operation are not new in the literature. Generally, the goals are the same as those stated here: improve the safety of the robotic vehicle while improving speed and mission effectiveness of the operator by allowing the robot to control those aspects it is most able to, while relying on human intelligence for difficult decisions and delicate operations.

The level of autonomy is often dictated by the complexity of the operating environment. For example, in many instances in simple environments, particularly for unmanned aerial vehicles, the sensing and processing on-board the robot can be used for transiting from one point to another without human intervention. However, for UGVs operating indoors or in urban environments, the amount of time that can be spent in an autonomous mode is limited.

In some semi-autonomous systems, autonomous modes or behaviours are used up until the point that either the system or the operator decides that it is performing poorly, and then the human operator takes over. Such a system may be referred to as supervisory control, or human-centered automation. However, when this type of system is being tele-operated, it is providing no assistance to the operator despite its on-board capabilities. A familiar example of this type of system might be an automated parallel parking feature on a passenger car, which completely takes over from the human when activated, and completely releases control once finished.

However, even when the operator is controlling the system, on-board intelligence and sensing can be used provide enhanced information to aid the human, such as providing warnings or more detailed graphical displays such as automatically generated maps of the environment or augmented reality. As an example, radars used on passenger cars may provide noises and lights when a driver attempts to change lanes with another vehicle in its blind spot.

In shared control systems, rather than having direct control over the robot mechanisms, a shared control system may filter or interpret operator commands based on its sensing and knowledge of the platform. The benefit of shared control is that the robot intelligence can do what it does best based on its rapid feedback loops and on-board sensing, while allowing the human to interpret the scene and decide what overall course of action of the system should take. A familiar example of this might be a collision avoidance system implemented on high end automobiles which will slow a vehicle down based on obstacles ahead, despite the operator having their foot on the gas pedal.

A shared control approach has been used for a variety of manned and unmanned systems. In manipulator systems, it may be used to control individual joint motions and avoid collisions while for mobile robots with complicated kinematics or dynamics it may be used to make the platform easier to drive. It has even been used to control motorized wheelchairs for the disabled, making them faster and safer. In a simple form, a shared control system may provide “guarded teleoperation”, which allows the robot to disable operator commands to protect itself from hazards. More advanced systems may not just provide safety, but also try to improve on operator commands based on what their intention most likely is.

The key to any shared control approach is successfully integrating the intelligence so that it is seamless to the operator, provides genuine assistance, and is not a nuisance in the performance of their tasks. It is also important that the system informs the user as to when and why it is modifying user input, and allows the operator complete override control when required.

1.3 Intelligent Shared Control of a Remote Robot (ISCORR)

The goal of the ISCORR project was to create driving assistance technologies that use on-board perception to detect and react to hazardous situations, improving operator performance. The target platforms were medium size man-portable platforms in common use for explosive ordnance disposal or chemical, biological, radiological applications in cluttered indoor or outdoor urban environments. The behaviours developed for this project include aspects of supervised autonomy, enhanced operator information, and shared control, as discussed in Section 1.2.
The project was funded through the DRDC Defence Industrial Research Program, and delivered by MDA – Brampton. The project ran from 2014 to 2016, and included 3 sets of field trials with Canadian Armed Forces operators and DRDC scientific staff.

The ISCORR behaviours provide a safety layer for the UGV, as well as automate simple tasks and provide improved operator situational awareness. The project developed the following:

1) A velocity feedback controller for a tracked platform with low control rate and inaccurate wheel encoders,
2) An obstacle detection and avoidance module based on 2D and 3D lidars that also provides proximity warnings, speed and heading adjustments, wall following, and doorway entering assistance,
3) A mapping and localization system for operator situational awareness and automatic backtracking after communications loss, and
4) An active track control and arm configuration system for stair climbing.

For most of the behaviours, the goal was to adjust operator commands with minimal interference (obstacle avoidance, wall following, doorway entry). These driving aids were meant to verify the operator’s commands against the motion of the robot, the robot’s mass distribution, and geometric information of the surroundings.

The stair climbing assist mode was also created which provides a shared autonomy interface for climbing stairs that would automate some aspects such as aligning with the stairs, maintaining heading, and arm positioning while leaving overall control of the robot to the human.

By necessity, the retreat from communications backtracking algorithm was completely automated.

Figure 1. Sensor placement on the Talon IV test platform
2. HARDWARE AND SOFTWARE

The implementation of the ISCORR system is intended to be modular, where some or all components can be integrated onto different robotic platforms with little parameter tuning. In this case, a QinetiQ Talon IV was used as a demonstration platform. Because of the limited space on the robot, sensor size and placement are crucial. The particular sensors were chosen because of their size and power consumption. An additional computer processing unit was also mounted on-board the robot. The sensor components, shown in Figure 1, are:

- A Hokuyo YVT-X002 3D lidar provides a 200 degree horizontal field of view and a 40 degree vertical field of view with 25m range at 20Hz. It was mounted on the lower manipulator arm to avoid blocking any of the operator cameras, while being able to provide data in a variety of manipulator positions, including stowed. Data from this sensor was used for obstacle detection during normal driving, as well as for aligning the robot to stairs for stair-climbing aid.
- A Hokuyo UTM-30LX 2D lidar is mounted on the rear of the vehicle facing backwards, with a 30m range and 270 degree field of view at 40Hz. It is used for obstacle detection during normal driving and scene mapping during retreat after loss of communication.
- The VectorNav VN-200 inertial measurement unit is mounted on the centre-top of the robot chassis, and provides robot orientation, velocity and acceleration at 30Hz. Its data is used for computing robot motion and balance during operation.

To avoid a complete hardware and software integration on the control station, the Talon LCU was retained to control the robot. However, an additional control laptop was added that completely takes over the LCU functionality, while adding the additional features required for this project. This structure is shown in Figure 2. The command messages and data exchange between the new operator control station and the robot are all implemented...
according to the Joint Architecture for Unmanned Systems (JAUS) protocol to maintain compatibility with other systems. The OpenJAUS library was used for this purpose. More details of this control structure are shown in Figure 3. The software onboard the robot was implemented on Ubuntu Linux, with modules in C++ and Matlab using ROS interfaces, while the operator control software was implemented in C/C++/.NET on a MS Windows platform.

![Robot Control Diagram](image)

Figure 3. Control and software architecture.

3. INTELLIGENT BEHAVIOURS

This section will discuss in detail the behaviours developed under the scope of this project. As discussed earlier, the goal was to provide driving assistance technologies using on-board sensing to improve operator effectiveness and allow them to focus on mission objectives.

3.1 Velocity Feedback Control

Many types of small military tracked robots including QinetiQ’s Talon IV are driven by controlling the current or voltage to the robot’s motors. In order to follow a particular path or trajectory precisely for avoiding obstacle and retreating after communication loss, it is necessary to drive the robot using velocity commands instead of the basic motor current or voltage command. The complete ISCORR system consists of a high-level vehicle velocity feedback control model employing a dual loop architecture as shown in Figure 4 to control the robot. The outer loop includes trajectory planning function and kinematics control functions which generate robot motion commands to follow the desired trajectory. The inner loop is a dynamics controller which tracks the velocity command given by the outer loop. Between these two loops, there is a safety function which can adjust the velocity command from the co-ordinate control functions to avoid colliding with obstacles or tipping over on an uneven surface.
The main vehicle control parameter is the vehicle speed limit which is applied to the velocity command from the outer loop. Several factors were considered when setting the speed limit:

- The dual loop control architecture requires the inner loop bandwidth must be much higher than that of the outer loop to avoid big position tracking and oscillation around the path.
- The linear speed must be limited to meet the requirement for stop distance and position accuracy.
- The rotational speed must be limited to make the simplified time invariant kinematics model valid.
- The curvature must be limited to reduce the coupling between the linear velocity and rotation velocity to simplify the dynamics control design.

The inner loop controller consists of a feed-forward controller and two independent PI controllers tracking the linear speed command and rotational speed command separately. The feed-forward controller is needed to deal with the nonlinearity in the system and the coupling between the rotational and translational motion using a priori knowledge of system dynamics. It also helps to achieve good speed control bandwidth without the instability risk from using high feedback gains. The PI controllers compensate the modelling error to ensure tracking accuracy.

### 3.2 Tip-Over Detection

A tip-over detection behavior was added to the system to assist drivers that may be unaware of the physical pose of the robot, providing a warning when certain limits are exceeded. The robot’s balance is estimated by computing the dynamic center of mass of the robot from the robot’s body mass distribution, current robot’s manipulator arm configuration, and current robot’s pitch and roll angles. To determine the tendency of the robot to tip over from the estimated robot’s balance, a Force-Angle (FA) algorithm is used to measure the angle of the total applied force on the center of mass of the platform with reference to the tipping over axis, which is derived from the ground contact points of the robot.

The final implementation uses gravity vector as the combined force in the FA algorithm and also assumes gentle robot motion on slopes and stairs. While the robot is in operation, it verifies its balance against two types of tipping over axis. The robot can roll over only around one side of the robot base rectangle when the robot stays on a rough terrain. Also, when the robot is on a steep slope or stair, it can tip over around the contact line between the robot and the inclined surface.
The GUI software application running on the ISCORR laptop computer notifies the operator of tip-over situation by displaying a warning icon at the middle of the screen, as shown in Figure 5. The seriousness of the situation is represented by using different icons.

![Figure 5. Modified QinetiQ GUI and tip-over indicators.](image)

3.3 Obstacle Avoidance

The ISCORR system uses a reactive obstacle avoidance algorithm to maintain robot safety in cluttered environments, based on current sensor data rather than a built up map of the environment. Details of this algorithm are presented in a companion paper to this one.

3.4 Localization, Mapping, and Retreat from Communications Loss

A major problem with tele-operation is the risk of loss of communications leaving the robot stranded by driving outside the communication range. This has a significant effect on the actions operators will take, as they may be hesitant to drive around corners or further away if video signal begins to degrade. This can be a major limitation on mission effectiveness.

As such, when the robot determines that the radio communication to the operator control unit has lost, the ISCORR system provides a retro-traversal capability in which the robot can return along its prior path autonomously until the communication is re-established, or until it reaches a user designated rally point along it prior path, greatly reducing the risk of a stranded system.

To support this, the robot needs to generate a map of the environment with the traversed path and also localize itself against the path while it is retreating. Several map building and robot localization algorithms (e.g. G-mapping, Karto) have been evaluated during the project. With limited processing power provided by the TALON’s on-board computer, a modified version of the ROS’s Hector-SLAM software fusing the lidar scans, IMU data and wheel odometry using an EKF filter is used in the final system to generate map and estimate the robot’s position and orientation. It is suitable for indoor environments where there are plenty of objects (e.g. walls, corners, hallways) as geometrical references and the floor is relatively flat. When the back-tracking behavior is initiated, the Hector SLAM algorithm first combines all the sensor data collected by the robot in the last few minutes or since the last safe location to generate a local 2D geometric map with the path traversed by the robot. The path includes information of the original robot poses and manipulator positions it should reverse for staying on the traversed track precisely. While the robot is retreating, the localization algorithm registers the current lidar scan to the map and estimate the robot’s position and orientation against the recorded path.

The initial version of the return home algorithm memorized all the paths the robot had traversed from the beginning of the mission, including those paths it had back-tracked while losing and re-acquiring radio signal. In certain situations,
this could result in a limbo situation. To avoid this problem, the current algorithm memorizes all the path sections in which the robot has retreated before and does not repeat them later, shortening the retro-traverse, and avoiding limbo situations.

The algorithm is executed as follows:
1) Record sensor data from the start of operation.
2) On loss of signal, build 2D map of the scene using data since the last safe location.
3) Generate a retreat path by re-localizing the robot through the map generated in step 2 using the odometry and the 2D lidar scans.
4) Begin to retreat, and while it is retreating, continue to localize the robot within the 2D map along the retreat path and also align the robot pose and manipulator position accordingly.
5) Stop the autonomous returning home behavior when one of the following conditions become true:
   a. Arrived at last safe location, or
   b. On user request. To be able to get the user request, the robot must be able to receive a strong signal from the command station.
6) If the motion is stopped from step 5b, find the nearest point to current position on the original forward path and remove the path section beyond the nearest point on the forward path.
7) Once the operator drives the robot manually again, append all new odometry and lidar scans to the end of the truncated forward path generated in step 6.
8) Repeat from step 2 when the robot runs into LOS again.

The retro-traverse behavior was also implemented with “return home” functionality. This uses the same method as for the retreat from communications loss, but allows the operator to set a home position for the robot to return to by operator command. This allows a robot to much more quickly retro-traverse all the way back to the operator at the end of a mission or to retrieve an object at a greater speed than operator tele-operation.

3.5 Stair Climbing Aid

One of the most dangerous and complicated tasks a robot operator conducts is stair climbing. This task is difficult not only because of the chance of the robot toppling if done incorrectly, but also because of track slip on stairs and the laborious sequence of manipulator and track movements that need to be conducted to keep the robot balanced. As such, the ISCORR stair climbing aid helps the operator automate portions of this task. The stair climbing aid provides these features as a sequence:

- Manual Drive – The operator has full control of the robot to drive it on stairs manually.
- Find Alignment – The robot system uses the on-board 3D lidar to scan the stair in front of it and then measures and corrects the angular alignment between itself and the stairs before mounting the stair.
- Get Ready – The robot system extends its manipulator arm forward to lower the overall center of mass and prepares for mounting the stairs.
- Climbing – The operator drives the robot up or down the stairs while the robot system tries to maintain its balance by correcting the commanded robot’s speed, heading, and also arm configuration during climbing. If the system senses that the robot is going to roll over or tip over, it will slow down the robot and notify the operator by displaying a warning on the screen.
- Dismount – The robot system reconfigures its manipulator and prepare it for landing at the end of the stair.

If the execution of a particular step fails (e.g. the robot gets stuck while it is climbing a stairs), the system will notify the operator through the GUI (e.g. the color of the corresponding button becomes red) and the operator can press the Manual Drive button to regain the full control of the robot at any time.

3.6 Stair Alignment Aid

One particular difficulty was sensing for stair alignment during initial alignment and during climbing. The most stable way for a tracked robot to climb a stair is to keep the robot moving in a direction which is perpendicular to the edges of the stair. This provides the maximum traction between the robot’s tracks and the stair’s surface, to avoid any side slipping and rolling. When the ISCORR’s stair climbing assistance is activated, the system uses the on-board IMU data...
to measure the angular misalignment between the robot’s direction and the stair’s edges, and adjusts the operator’s
command to realign the robot with respect to the stair gradually.

To find the direction of the stair or stair’s edges, the system relies on the 3D point cloud of the stair acquired by the
front-facing 3D lidar. The 3D point cloud of the stair in front of the robot is first orthogonally projected to the floor (X-Y)
plane along the downward (Z) direction based on the robot arm’s configuration. Assuming the width of the stair is at
least half of the width of the robot’s foot print (roughly 35cm), a probabilistic Hough transform is then applied to the
projected image and long line segments are then extracted from the image (see Figure 7).

Parallel lines extracted from the image are potential candidates of the stair’s edges. The orientation of the stairs with
respect to the 3D lidar frame is solved by extracting the parallel lines from the image. Without loss of generality, the
problem can be simplified to find three longest line segments that are also parallel to each other, which is equivalent to a
maximum a joint probability function. The parallel component of the joint probability density function was modeled as a
zero-mean Gaussian with 3 degrees standard deviation. Relying on the rich 3D information provided by the 3D lidar, this
algorithm requires less computing power than many of the existing stereo based and 2D lidar based stairs alignment
techniques.

However, finding the orientation and position of a staircase when the robot is about to descend or descending down the
staircase is not as trivial as ascending a staircase since any sensor mounted on the robot can only see the treads but not
the risers of the staircase while the robot is descending with all its sensors looking downward. In this problem, the stair
alignment algorithm must rely on the partial view of the staircase provided by the sensor to recognize the treads and then
determine the orientation of the staircase from the step edges.

For this situation, a version of the Difference of Normals (DoN) approach is used for finding the orientation of the
descending stair. The general concept of this algorithm is that for each point in a given 3D point cloud, compute two
point normals from two regions around the point with different radii. Then the vector difference between these two
normal represents the DoN of this point. The DoNs of those points which belong to the surface of the treads should be
small, while the DoNs of those which lie on the step edges or correspond to the discontinuity between two consecutive
steps should be large (see illustration in Figure 6). Then, similar to finding orientation of the ascending stair, a Hough
transform line extraction is used to extract the stair edges from the DoN image.

![Figure 6. Difference of Normals used to locate stair axis.](https://www.spiedigitallibrary.org/conference-proceedings-of-spie)
4. USER TRIALS AND FEEDBACK

The ISCORR system was developed using an iterative design process, with behaviours tested at each of three operator trials conducted before further development was pursued. Participants for the trials consisted EOD operators for various units of the Canadian Armed Forces, Toronto Police Services, as well as DRDC scientific staff. Field trials #1 and #2 were limited trials conducted in an office environment in 2015, with each behavior tested independently. The third trial was conducted in an abandoned school in 2016, with a more complete scenario that involved starting on the first floor, driving around, climbing to a second floor, finding a specific item, and returning to the start position. Trial results are presented below under the relevant behaviours.

4.1 Obstacle Avoidance

The obstacle avoidance algorithm was tested in Field Trials #2 and #3 by running through obstacle courses in an office environment with boxes, chairs, hallways, doors, etc. In this scenario, the obstacle avoidance behavior produces some implicit behaviours that greatly sped up robot operation, reducing operator time by approximately 30-70%. Examples include driving down long hallways or entering doorways, where the driver did not need to concentrate on aligning the robot, and could just press the stick forward with the robot self-centering. The obstacle avoidance also can help position the robot near to tables or cabinets for manipulation tasks.

Feedback from operators indicated a number of relevant points:

- A GUI display of the local surroundings detected by the lidar could greatly assist the operator with situational awareness, even without the presence of a behavior to control the robot.
- Sensor placement is difficult on a robot with a manipulator. The current position on the lower manipulator arm is acceptable, but can create blind spots.
- Hallway navigation with obstacle avoidance was highly valuable for both novice and experience users.
- Doorway navigation with obstacle avoidance was highly valuable for novice operators, but less so for experienced operators. Without proper tuning it can be intrusive by disallowing tight turns from the side of a door, and works best with a more straight-on approach.
- It is expected that the collision avoidance could be used with very small safety margins, as usually minor collisions with doorways or obstacles are not of major concern.
- Operators specified they would use the collision avoidance when entering a scene, but would turn it off when in close proximity to a suspicious device or object requiring investigation.
- It would be useful to provide feedback on the GUI to indicate when and how much the robot is adjusting commands.

Figure 7. Lidar points clouds and step edges extracted from Difference of Normals method.
• The collision avoidance features should be somewhat tuneable, so that it could still be useful when carrying an object in the manipulator claw.
• A quick override feature is desired so that the obstacle avoidance can be quickly disabled for a short amount of time. An example might be pushing the joystick forward twice to indicate a desire to move forward despite the collision avoidance, without having to disable then re-enable the behavior.
• During tele-operation around obstacles beyond line-of-sight, many of the corrections provided by the system were actually transparent to the operator.

Further results of the obstacle avoidance algorithm are presented in the companion paper on that topic.

4.2 Mapping and Loss of Communications
Retro-traverse was demonstrated a number of times in the field trials, and did not fail to retreat from communications loss, including through doorways, around obstacles, etc. However, it would probably be possible to design a scenario where it could fail, such as extremely cluttered environments or soft terrain, etc. Figure 8 shows some typical scenarios in which this functionality was tested. Test results show that the modified algorithm can generate a consistent local map and localize the robot within the robot’s worksite for hundreds of meters of travelling distance from the starting location.

![Figure 8. Examples of mapping and returning home in an indoor environment. The red line is the path travelled by the robot before communications loss, and the green line is the automatic retro-traverse.](image)

Again, feedback from operators indicated a number of relevant points:

• The reverse motion performed by the robot during loss comm and retreat needs to be extremely accurate to avoid collisions. The robot should reverse the exact traversed path with the same robot poses and manipulator positions (with near to zero positional error) during retreat.
• There will be an issue on whether stowing the robot arm or not before retreat. If the gripper is holding an object then it will not be possible to stow the arm without damaging the object or the robot. However, the robot may not be able to retrace the path with an extended arm without colliding into objects.

• Communication loss is most likely to occur when the robot is moving, while the robot is generally stationary for manipulation tasks. Therefore, the retreat function should always be disabled before any manipulation tasks.

4.3 Stair Climbing

Tests of this behavior were conducted in all three field trials. In particular, tests were conducted using various levels of autonomy, including:

• Full manual operation, with operator line of sight to the robot,

• Full manual operation, without operator line of sight to the robot,

• Arm position aid where the operator controls the speed and direction, and selects between pre-defined manipulator arm positions and motions for handling different situations,

• Stair alignment aid where the operator controls the speed and arm positions, while the robot keeps itself aligned to the stairs, and

• Near fully autonomous where the robot controls the speed, direction and arm configuration, with little control to the operator, except for transitions between climbing conditions (i.e. aligning, mounting, climbing, dismounting).

Although the autonomous behavior did speed up the conduct of ascending and descending stairs, the best results were found with only the stair alignment aid functional. Results are presented in Figure 9 and Figure 10.

![Figure 9. Time to complete stair ascending with different methods.](https://www.spiedigittalibrary.org/conference-proceedings-of-spie)
Again, the following points were presented by the operators:

- The stair climbing aid is particularly useful for the Talon robot, but it may not be as useful for other robots that require less attention (i.e. with lower center of gravity, longer base, with flippers or without manipulators).
- The stair climbing aid is somewhat dependent on the available space of the approach and landings of the stairways, and may not be useful in very tight stairwells.
- The behaviour should provide a notification of the stair’s pitch angle in advance, to indicate whether they are actually traversable by the robot or not.

5. CONCLUSION

This project developed a number of practical behaviours to assist in the situational awareness and mission effectiveness of robot operators in indoor and urban environments. Two particular modules were found to be particularly useful. Firstly, the collision avoidance module was able to speed up operators by assisting in obstacle avoidance, wall following, door entry, and robot positioning. Secondly, a “retro-traverse” module was able to effectively implement robot mapping to allow a robot to recover from communications loss in complex environments, as well as to speed up the recovery of objects from the environment. Even in manual tele-operated modes, it was found to be useful to exploit on-board sensors to provide information to the operator, such as tip-over warnings, obstacles in the environment, and 2D maps.

As for the stair climbing behavior, complete automation of this task is not particularly effective, and does not provide a significant time benefit over manual tele-operation. The ability to sense the size, angle and position of stairs was effectively demonstrated. However, the actually configuration and positioning of the robot arm is highly dependent on the robot and the structure of the staircase itself. With that being said, having the robot maintain heading on the stairs despite slippage does provide significant assistance to the robot operator.

This project has demonstrated that it is very feasible to integrate autonomous behaviours to improve mission effectiveness with reasonable cost and size of sensors.
6. ACKNOWLEDGEMENTS

The funding for this project has been provided by DRDC Defence Industrial Research Programme and MacDonald Dettwiler and Associates. QinetiQ North America provided access to their Talon IV robot and modified the software to enable easy integration with ISCORR system. Members of Canadian Forces and Toronto Police Services operated the system during field trials.

REFERENCES


## Safeguarding Autonomy Through Intelligent Shared Control

### Authors

Giesbrecht, J.; Ng, H.-K.; Zhang, M.; Tang, J.; Bondy, M.; Jasiobedzki, P.

### Date of Publication

September 2017

### No. of Pages

16

### No. of Refs

25

### Descriptive Notes

External Literature (N)

### Sponsor's Activity

DRDC – Suffield Research Centre  
Defence Research and Development Canada  
P.O. Box 4000, Station Main  
Medicine Hat, Alberta T1A 8K6  
Canada

### Originator’s Document Number

DRDC-RDDC-2017-N023

### Future Distribution

Public release
This paper overviews the development and operator testing of a shared autonomy system for small unmanned ground vehicles operating in indoor environments. The project focused on creating driving assistance technologies to reduce the burden of performing low-level tasks when operating in cluttered or difficult areas by sharing control between the operator and the autonomous software. The system also provides a safety layer to prevent the robot from becoming disabled due to operator error or environmental hazards. Examples of developed behaviours include obstacle proximity warning, centering the vehicle through narrow doorways, wall following during long traversals, tip-over indicator, stair climbing aid, and retreat from communications loss. The hardware and software were integrated on a QinetiQ Talon IV robot and tested by military operators in a relevant environment.

Shared control, semi-autonomous, obstacle avoidance, stair climbing, robot control