Cite as: Alejandro Catala, Alejandro Moreno. Smart Navigation for a Storytelling Multi-Robot Setting. IROS 2018 Second Workshop on Multi-robot Perception-Driven Control and Planning, 3 pages, 2018.

Alejandro Catala, Alejandro Moreno

Smart Navigation for a Storytelling Multi-Robot Setting*

***Abstract*— This paper reports the design and implementation of a navigation planner for a multi-robot system intended for collaborative storytelling. It supports multiple users to deliver navigation plan requests over multiple robots, which can include high-level motion behavior.**

I. INTRODUCTION

The use of robots as experimental learning tools in school settings is becoming increasingly popular within the education community due to the reported positive effects and the open horizons that it can have in enhancing the learning experience (e.g. [9][7]). While many of these proposed solutions employ static robots, we are slowly moving towards the design of robots with limited navigational capabilities that can be programmed by children in the context of STEM activities through Scratch extensions or similar block languages [12][1]. Naturally, the addition of programmable robot motion can enable designers to create more complex and, ultimately, more beneficial experiences for children. However, previous observations in our own research [3] suggest that motion in robotic learning settings can cause children to (1) allocate much of their cognitive resources to control the robot due to the number of low-level commands that are required for fine-grained user control (e.g. move forward, stop, turn left, etc.), and (2) disconnect from the designed (learning) experience. In both cases, reaching the desired learning goals/outcomes can become difficult or even impossible. For instance, in a live storytelling application in which children have to dynamically create and play out a story, controlling the movement of the robots at such a low-level diminishes the focus on the storytelling process and enhances how fun driving robots is.

This is in-and-of itself an important problem, but when considering that the community is slowly transitioning towards related research in multi-robot settings [10][8][2][6], solving it becomes of critical importance. Thus, in this paper, we propose a potential solution to the problem by facilitating the manner in which robots are controlled by children. More specifically, we report the design and implementation of a high-level planner using reciprocal collision avoidance (ORCA) [13] for a multi-robot setting supporting collaborative storytelling (see Figure1).

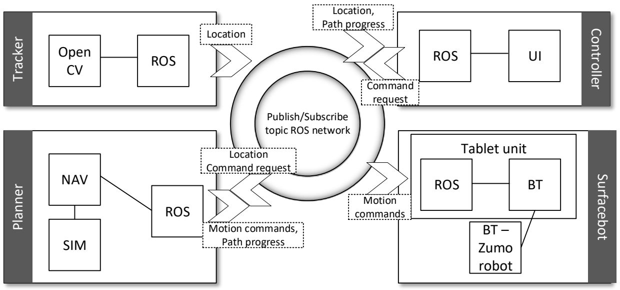
II. SYSTEM OVERVIEW

A. Requirements

There are several requirements to be covered in the context of our project that impact directly on the navigation features. The most important three are Safety, Multiplicity and Autonomy. Safety means providing methods to prevent robots from falling off the table by accident and preventing them from colliding with each other or with obstacles. Multiplicity is supporting multiple users and multiple robots in the same interactive setting. Finally, Autonomy refers to providing assistance by means of high-level commands rather than low-level, diminishing users' workload as discussed earlier.

Figure 1. Two robots in our storytelling setting.

Figure 2. Architecture components.



B. Architecture and Implementation

Our system is designed in a modular fashion, and is composed of 4 modules (Figure 2). This not only facilitates testing, but also makes replacing the implemented components in the future easier. The components are integrated through ROS by sharing information in the overlay network.

A video associated to the submission can be found at: https://www.youtube.com/watch?v=Gf7TWT4WAmo

---

* A. Catala was at the Human Media Interaction group of the University of Twente. He is now with the Research Center on Information Technologies of University of Santiago de Compostela (CiTIUS), Spain (phone: +34 8818 16460; e-mail: alejandro.catala@usc.es).

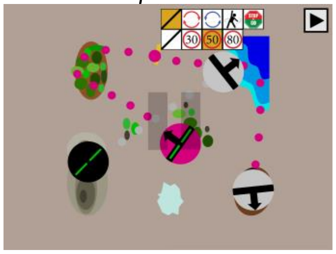
A. Moreno is with the Creative Media and Game Technologies group at Saxion University of Applied Sciences, Netherlands. (e-mail: a.m.morenocelleri@saxion.nl).

Surfacebot: The hardware of the robots in our setting actually consists of a set with a robotic unit and a tablet. The tablet is just a 7-inches Android tablet embedded into a plastic case and the base is a Zumo robot for Arduino by Pololu[2] expanded with a Bluetooth shield, which results in a non-holonomic two-wheeled robot. Regarding motion functionality, the tablet app uses the Bluetooth link to send low-level motion commands to the microcontroller that controls the motors. The case has a unique fiducial marker that is used to track the position of the robot.

Tracker: As robots do not have on-board instrumentation for solving positioning, we opted for implementing a tracker based on fiducial ArUco markers [4]. This C++ ROS node is responsible for publishing the location and rotation of the Surfacebots along with timestamps into the ROS topic. We used a Logitech C920 webcam which provides top-down 1920x1080@30fps RGB images. A virtual coordinate system is automatically created by placing two special markers (origin and limit) in opposite diagonal corners of the playing field (roughly $4m^2$ in our setting). The rotation of a robot is determined by the difference between its marker rotation and the camera, whereas the position is given within the created virtual coordinate system. The relative position of the robot marker with regard to its center is modeled, making it possible to have different offsets for each robot. The loosely-coupled design of our architecture would make it possible to replace this component with on-board positioning subsystems.

Controller GUI app: This component has a Graphical User Interface implemented in a touch-enabled Android app as seen in Figure 3. It gives a view of the tracked system, allowing users to manipulate robot behaviors and motion plans. Multiple controller apps can be connected to the system, enabling multiple users to control either the same or different robots. Users can instruct robots to move in two different ways: tapping a desired destination, and drawing a path for the robot to follow. Paths can be personalized by adding behavior modifiers which are triggered at specific locations in the path. These modifiers can be speed modifiers changing the travel velocity, storytelling specific behavior modifiers such as speech, visual and emotional behavior created with other tools, and motion modifiers that implement specialized movement commands that expand the navigation capabilities. For example, it can implement arbitrary rotations or in-place movements that have priority over navigation and must be handled in this way because they entail a variation on the path control.

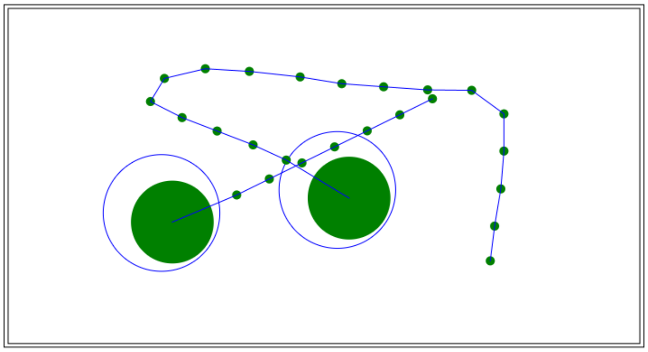
Figure 3. Screenshot of Controller GUI tablet app.



A remarkable feature of our system is that it supports both live and batch mode. In live mode, plans are sent to the planner as soon as input touch commands are carried out (e.g. a path or a target are given to a robot). In batch mode, the plans are sent only under request, facilitating users to design plans for an arbitrary number of robots on screen and executing them simultaneously.

Planner: This core component is implemented as a Java ROS node. It listens to the locations and rotations, which in our current implementation come from the tracking subsystem, and the plans requests coming from the multiple controller GUI apps that dynamically may have joined the system. It uses this information to run a simulation and calculate the velocity vectors that will allow each robot to follow their paths without colliding into each other. This is then pushed to the robot, to actuate it to move in the calculated fashion. The plans requests are processed as follows. Each plan is about a single robot. If multiple plan requests are about the same robot, the last received request will prevail. So conflicts between multiple users should be solved by relying on social protocols rather than technology, which is an acceptable solution as there is visible immediate feedback. For target destination requests, a single path between the current and target locations is created. For path requests, the path is broken into smaller segments by sampling it at regular intervals, resulting in sub-paths that sequentially make up the entire path (see Figure 4). Modifiers are considered during this process so that they are either at the end or start of a sub-path. The navigation planner will drive the robots towards the first path element in their corresponding paths by setting the vector velocity for each robot. This is carried out by integrating the navigation algorithm based on *optimal reciprocal collision avoidance*[3] [13] subject to differential-drive constraints as formulated in [11]. With this approach, we treat the navigation as an interactive and dynamic process in which the goal is not just to reach a final destination but may include showing motion behavior while following a trajectory. As a consequence, we do not strictly plan each path from begin to end, but the planner holds a set of intermediate sub-paths that will be followed while avoiding collisions and responding dynamically to registered obstacles.

Figure 4. Navigator view.



Algorithm 1 shows the navigation loop carrying out the simulation and taking inputs into account. When setting the scenario, we are placing virtual boundaries around the periphery of the tracking area defined by the origin and limit markers. Additionally, dynamic obstacles are being handled by adding non-motorized agents with a bounding circle, and

---

[2] Pololu Zumo Robot: https://www.pololu.com/product/2510

[3] http://gamma.cs.unc.edu/RVO2/

so taking part of the simulation as any other robot with no velocity.

Two high-level motion behaviors, social avoidance and following peers, were implemented by checking constraints and adding plan goals to the robots involved. In this way, we can just rely on the collision avoidance algorithm. Additionally, there is a fatigue model implemented to control the depletion and the recovery of surfacebot's virtual energy, which can be activated by the designer to prevent children to continuously driving the robot without focusing on the storytelling activity.

Algorithm 1: Navigation loop

```
Setup ROS topics
Setup Scenario
Do
   Sleep based on processing frequency
   Add new agents, if any
   Update tracked agent
   Apply Kalman filter on rotation
   Process command requests
   Calculate Preferred Velocities based on next subgoals
   Carry out simulation Step
   Set Wheel Velocities and send out movement commands
Until navigation is stopped
```

III. CONCLUSIONS AND FUTURE WORK

We have presented a high-level navigation planner for a multi-robot system suited to collaborative storytelling. It relies on the ORCA formulation found in the literature in order to supply collision-free navigation. We have developed several additional features capable of delivering social motion behaviour in such way that it frees users from the fine-grained control interactions that would have a negative effect on the main goal of the ongoing user activity.

For future work, there are several possible avenues for improvement. We have noticed that the adjustment of algorithm parameters is sensitive. For instance, the kinematic model used for the differential-drive robots needed an enlarged radius and may therefore lose maneuverability if the available space is not large enough for the number of robots involved. Additionally, the characteristics of the actual setting and physical robots used can affect the speed and the rotation torque, and the grip with the surface. This is critical in settings that have to be easy to deploy. Thus, we need to consider models fitting the real shape [5] and methods to automatically calibrate the physical parameters.

The management of dynamic obstacles is handled by modeling non-motorized agents that can only be moved by forces outside of the control of the system. Considering more complex shapes than simple bounding circles would give more freedom to design scenarios and applications. The interaction between the navigation provided and the special motion behavior can be studied specifically. For example, when a robot is rotated in place, imperfections in the physical system can lead to unintended forces and movements that differs from ideal or simulated rotations. It could trigger some other robot to move because the responsibility of avoiding collisions is shared between robots. In applications with similar requirements to us, this can be certainly troublesome and annoying for users but could be addressed by implementing a priority scheme between robots and/or type of motion goals.

ACKNOWLEDGMENT

The core work has been carried out at University of Twente. Research partially funded by H2020 MSCA-IF grant No. 701991 coBOTnity, European Regional Development Fund and Consellería de Cultura, Educación e Ordenación Universitaria (acc.2016-2019, ED431G/08). Special credits go to R. Campisi, L. Sandfort, W. Timmermans and W. van Veelen for contributing to this research project while working in their group assignment.

REFERENCES

[1] Bellas F. et al. (2018) The Robobo Project. Bringing Educational Robotics Closer to Real-World Applications. In: Lepuschitz W., Merdan M., Koppensteiner G., Balogh R., Obdržálek D. (eds) Robotics in Education. RiE 2017. Advances in Intelligent Systems and Computing, vol 630. Springer, Cham.

[2] A. Catala, M. Theune, H. Gijlers, D. Heylen. Storytelling as a creative activity in the classroom. In: Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition (C&C '17), pp. 237-242, ACM, New York, NY, USA (2017).

[3] A. Catala, M. Theune, D. Reidsma, S. ter Stal, D. Heylen: Exploring children's use of a remotely controlled surfacebot character for storytelling. In: Chisik Y., Holopainen J., Khaled R., Luis Silva J., Alexandra Silva P. (eds) Intelligent Technologies for Interactive Entertainment (INTETAIN 2017). LNCS, vol 215. Springer, Cham (2018).

[4] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, M.J. Marín-Jiménez, Automatic generation and detection of highly reliable fiducial markers under occlusion, Pattern Recognition, Volume 47, Issue 6, 2014, Pages 2280-2292, ISSN 0031-3203.

[5] A. González-Sieira M. Mucientes A. Bugarín. (2016) An Adaptive Multi-resolution State Lattice Approach for Motion Planning with Uncertainty. In: Reis L., Moreira A., Lima P., Montano L., Muñoz-Martinez V. (eds) Robot 2015: Second Iberian Robotics Conference. Advances in Intelligent Systems and Computing, vol 417. Springer, Cham

[6] R., Grieder, J. Alonso-Mora, C. Bloechlinger, R. Siegwart, P. Beardsley. (2014). Multi-robot control and interaction with a hand-held tablet. In *ICRA 2014 Workshop on Multiple Robot Systems. IEEE*.

[7] J. M. Kory Westlund, H. Won Park, R. Williams, C. Breazeal. 2018. Measuring young children's long-term relationships with social robots. In Proceedings of the 17th ACM Conference on Interaction Design and Children (IDC '18). ACM, New York, NY, USA, 207-218.

[8] A., Krzywinski, W. Chen: Hi Robot: Evaluating Robotale. In: Proceedings of the 2015 ACM International Conference on Interactive Tabletops and Surfaces, ITS 2015, pp. 367-372, (2015).

[9] I. Leite, M. McCoy, M. Lohani, D. Ullman, N. Salomons, C. Stokes, S. Rivers, B. Scassellati. 2015. Emotional Storytelling in the Classroom: Individual versus Group Interaction between Children and Robots. In Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction (HRI '15). ACM, New York, NY, USA, 75-82.

[10] A. Özgür, S. Lemaignan, W. Johal, M. Beltran, M. Briod, L. Pereyre, F. Mondada,, P. Dillenbourg. 2017. Cellulo: Versatile Handheld Robots for Education. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction* (HRI '17). ACM, New York, NY, USA, 119-127.

[11] J. Snape, J. van den Berg, S. J. Guy and D. Manocha, "Smooth and collision-free navigation for multiple robots under differential-drive constraints," 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, 2010, pp. 4584-4589.

[12] S. Magnenat, J. Shin, F. Riedo, R. Siegwart, M. Ben-Ari. 2014. Teaching a core CS concept through robotics. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (ITiCSE '14). ACM, New York, NY, USA, 315-320.

[13] J. van den Berg, S.J. Guy., M. Lin, D. Manocha. (2011) Reciprocal *n*-Body Collision Avoidance. In: Pradalier C., Siegwart R., Hirzinger G. (eds) Robotics Research. Springer Tracts in Advanced Robotics, vol 70. Springer, Berlin, Heidelberg.