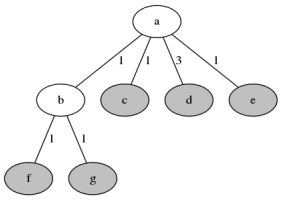
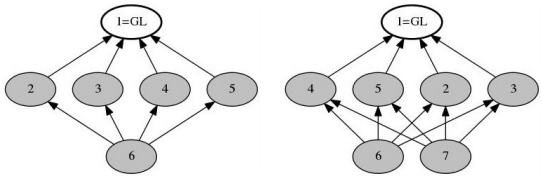


Figure 1: An example of a tree reconstructed from the set of the leaves (c, d, e, f, and g). The leaf vertices represent the visible robots from the perspective of some other formation robot, which is not part of the tree. The other vertices represent other robots in the formation, non-detectable by the sensors of that robot. The input tree distance matrix is consistent with the distances between the leaves, e.g. $dist_T(d, f) = 5$ and $dist_T(c, e) = 2$.

The ULS and the UVLS algorithms use the tree-reconstruction algorithm to reconstruct the current leader-follower relations between the formation robots, starting from the observed set (represented as leaves in the tree), on to their local leaders and to the next local leaders, and so on, until the root, which is assumed to be the GL (or the robot that is the the least common local leader of the part of the formation spawned by the visible set). ULS is used to select the most accurate local leader from the currently visible set, while the UVLS takes into account the entire visible set at given time to construct a virtual local leader. Naturally, the process is repeated for every incoming observation, with some form of smoothing to reduce the possible large effect of outlined observations.

3 RESULTS

We present a partial set of results that were obtained from simulations of a formation moving in a straight line, with six or seven robots in a formation, and vision-based state estimations made by the robots. The experiments were performed using the ROS/Gazebo simulator[1][2], simulating the behavior of Hamster robots[3].



(a) 6 robots formation (b) 7 robots formation

Figure 2: An example of the formation patterns and the robots' ability to sense other team members used in the experiments. Here, robot 1 is the GL and the local leader of robots $2-5$, and robot 6 in 2(a) (and also 7 in 2(b)) can choose any one of robots $2-5$ to follow (or a virtual combination of a subset of them).

The experiments compared ULS and UVLS to a centralized approach, where there is perfect communication between the GL and the other robots (the GL serving as the central computational unit, in addition to moving the formation). GL checked the noise level (mean) between every pair of robots (as was reported by the noise simulation unit) and assigned the local leaders to all robots in the team. In all experiments, robot #5 did not have an over time increasing noise model and the noise of its sensor was dependent on the distance and angle to the observed robot only. The centralized approach selected it as the best local leader for the robots that could sense it in the formation and the expectation was for ULS to converge to the same selection.

The true distances between each robot and the global leader were recorded at 5Hz rate during the execution and the overall formation error was calculated as the norm of the vector of deviations of each robot from its expected distance to the GL. Similar experiments results were averaged over the time axis to compare between the performance of every method that was put into test.

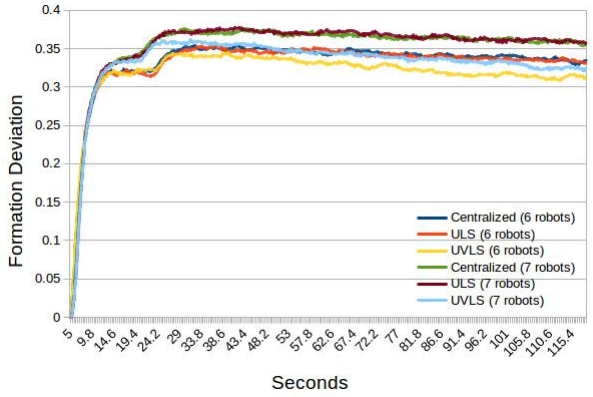


Figure 3: Average formation deviation over time (lower values correlate to more accurate formation control, i.e., better performance). The actual movement always began at the 5^{th} second of the experiment to overcome the asynchronous model loading in Gazebo.

As can be seen in Figure3, ULS algorithm performed similar to the centralized approach (statistically indistinguishable), while UVLS was statistically significantly better. It is important to note that the virtual local leader accuracy depends on the correlation between the deviations of the state estimations (made by the same robot) and UVLS would not necessarily perform better than the other two algorithms in case of strong correlation between the errors (e.g. if a robot tends to overestimate the distances, or the angles, to the same direction).

4 FUTURE WORK

For future work, we plan to extend this work by providing empirical results performed on real robots, while modeling non-holonomic estimation errors, and a more complex navigation scenarios in both simulation and real world.

Additionally, since the main contributor to the runtime complexity of both algorithms is the tree-reconstruction algorithm, there is an ongoing work to replace it with a more efficient reconstruction algorithm, that is not based on `NeighborJoining` .

[1]http://ros.wiki.com
[2]http://gazebosim.org
[3]http://www.cogniteam.com/hamster4.html

REFERENCES

[1] Javier Alonso-Mora, Eduardo Montijano, Mac Schwager, and Daniela Rus. 2016. Distributed multi-robot formation control among obstacles: A geometric and optimization approach with consensus. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 5356–5363.

[2] Tucker Balch and Ronald C Arkin. 1998. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation* 14, 6 (1998), 926–939.

[3] Jakob Fredslund and Maja J Mataric. 2002. A general algorithm for robot formations using local sensing and minimal communication. *IEEE Transactions on Robotics and Automation* 18, 5 (2002), 837–846.

[4] Gal A Kaminka, Ilan Lupu, and Noa Agmon. 2018. Construction of Optimal Control Graphs in Multi-robot Systems. In *Distributed Autonomous Robotic Systems*. Springer, 163–175.

[5] Gal A Kaminka, Ruti Schechter-Glick, and Vladimir Sadov. 2008. Using sensor morphology for multirobot formations. *IEEE Transactions on Robotics* 24, 2 (2008), 271–282.

[6] M Anthony Lewis and Kar-Han Tan. 1997. High precision formation control of mobile robots using virtual structures. *Autonomous robots* 4, 4 (1997), 387–403.

[7] Naruya Saitou and Masatoshi Nei. 1987. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution* 4, 4 (1987), 406–425.

[8] James A Studier, Karl J Keppler, et al. 1988. A note on the neighbor-joining algorithm of Saitou and Nei. *Molecular biology and evolution* 5, 6 (1988), 729–731.