Proceedings Paper:

Lambert, ED orcid.org/0000-0002-2297-0441, Romano, R orcid.org/0000-0002-2132-4077 and Watling, DP orcid.org/0000-0002-6193-9121 (2020) Simulating Decentralized Platooning for Coordinated Conflict-Free Motion of Mobile Robot Fleets. In: 2020 3rd International Conference on Robotic Systems and Applications Proceedings. 2020 3rd International Conference on Robotic Systems and Applications (ICRSA2020), 14-16 Jun 2020, Online. . ISBN 978-1-4503-8764-4

https://doi.org/10.1145/3402597.3402603

Simulating Decentralized Platooning for Coordinated Conflict-Free Motion of Mobile Robot Fleets

Edward Derek Lambert
Institute for Transport Studies
Leeds, UK
tsedl@leeds.ac.uk

Richard Romano
Institute for Transport Studies
Leeds, UK
R.Romano@leeds.ac.uk

David Watling
Institute for Transport Studies
Leeds, UK
D.P.Watling@its.leeds.ac.uk

ABSTRACT
Intersection platooning has been shown in several studies to provide higher throughput and hence greater productivity for material transfer using fleets of mobile robots, however questions remain about the behavior in a full range of traffic situations as no algorithm has been proven complete. As a result a lower throughput backup system is typically used in practice. A state of-the-art method for decentralized platooning based on an intersection controller has been implemented. A number of test cases are described to probe the performance limits. Furthermore, an outline of a fleet control system which uses only platooning for coordination is described. This will allow edge cases to be investigated in simulation.

CCS Concepts
•Computer systems organization~Embedded and cyber-physical systems~Robotics~Robotic control •Computing methodologies~Artificial intelligence~Planning and scheduling~Multi-agent planning

Keywords
Intersection Platooning, Mobile Robot Fleet, Conflict-Free Motion, Simulation

1. INTRODUCTION
Coordinated conflict-free motion of a number of mobile robots in order to complete a material transfer task is important in the operation of fleets of AGV (Autonomous Guided Vehicles) used in flexible manufacturing and automated warehouses [18] and [7]. A crucial sub-problem is conflict resolution between multiple AGVs, without control of task assignment or scheduling.

AGV motion coordination can be posed as a variation of the Multiple Vehicle Routing Problem with the addition of challenging spatio-temporal constraints, preventing collisions between each vehicle, as well as the usual timing and capacity constraints [13]. In [11], solutions are classified into centralized, decentralized and decoupled approaches. Each approach may be either optimal or heuristic based on whether or not they find the global minimum of some objective function.

ICRSA 2020, June 14–16, 2020, Chengdu, China

ACM ISBN 978-1-4503-8764-4/20/06…$15.00
https://doi.org/10.1145/3402597.3402603

It is also possible to classify approaches based on the limitations they place on the state space of each AGV. Most practical methods incorporate both obstacle avoidance constraints and differential motion constraints into some sort of roadmap. This is often a graph with vertices at key points in the reachable state space, connected by edges representing feasible motion between them. The effect of increasing resolution on path quality (measured by reduction in the length of the shortest path) and computation time are studied in [17].

An optimal decoupled method operating on graphs is presented in [19]. Optimal conflict-free motion is posed as a large Integer Linear Program. Resolution complete general purpose algorithms are used to solve it for 150 robots in just over 10 seconds. The lattice/graph construction has recently been developed further to ensure kinematic constraints are met and improve coverage of the state space around obstacles [20]. In [13], the combined problem of DCFVRP (Dispatch and Conflict-Free Vehicle Routing Problem) for flexible manufacturing is formulated as an integer program and two different decoupled algorithms are presented to solve it: local search and random search. Neither of the proposed algorithms is complete but local search found more valid solutions in the 10 random examples tested, all involving three vehicles.

Decentralized control is another option to decompose large scale problems which take too long to solve centrally [2]. Although limiting the information available to each decision maker can make reasoning about collective behavior more difficult, various attempts to decentralize conflict-free routing have been made. In the field of conflict free routing for mobile robots, [8] presents a solution which generates a graph representation of the free space - effectively a roadmap - with the property of 'collision avoidability.' This means that every node on the critical path must be at most one move away from a node that does not obstruct the critical path. The critical path is defined as the union of all the shortest paths between pick/drop locations in the roadmap. During decentralized planning, AGVs attempt to reserve 'private zones' consisting of the node on the critical path along with all adjacent collision avoidance nodes. Each AGV has an identical roadmap, plans the shortest path to its own goal and negotiates with those nearby based on a numeric priority to reserve the nodes in its own path. An AGV requests those in its path move to their collision avoidance node, and those with a lower priority will do so. Proof is given of correctness, that deadlocks are avoided, but throughput is sub-optimal with low priority vehicles frequently forced to stop and wait. An alternative decentralized solution, based on a roadmap with two levels of detail is summarized in [15]. Conflict-free routing primarily takes place at the most detailed level, based on prioritized roadmap reservation with local negotiation to guarantee correctness [5]. In [4], the speed of the approaching AGV is optimized at each intersection in a similar way to centralized intersection platooning. The result is higher throughput as time consuming negotiation is avoided in most cases.

Intersection control, based on platooning, is a concept developed for the operation of anticipated CARVs (Connected and Autonomous Road Vehicles). A recent review of approaches for intersection and merging coordination is given in [14]. Centralized optimization approaches improve on early ideas like First-Come-First-Served spatial reservation from [10] by minimizing fuel consumption, but the rapid increase in state space with larger numbers of vehicles will need to be addressed before large scale adoption. The communication channel connecting every vehicle with the central controller introduces a single point of failure, the reliability effect of which is difficult to evaluate in existing simulations. Moreover there are few CARVs currently available making a practical experiment unfeasible in most cases. Attempting to address these limitations are decentralized methods such as fuzzy-logic, virtual vehicle platooning and invariant set approaches.

Recently [16] described an approach to the DCFVRP for flexible manufacturing based on dynamic platooning with vehicle-to-everything messaging and consensus speed control, resulting in a decentralized heuristic solution with some additional rules to ensure correct behavior and avoid deadlocks by adding a reservation protocol for some parts of the roadmap. This was combined with feedback from the queue length at different workstations in a traffic management heuristic. Simulation results show an impressive improvement compared to the first-come-first-served scheduling approach meant to represent industry standard practice.

It is shown in [4] that approach-speed choice by a centralized intersection controller provides higher throughput compared to stopping and giving way for decentralized reservation negotiation. The speed choice is shown to be optimal if a solution is found, however it is not shown to be complete. This suggests there are certain roadmap and traffic combinations where there is no solution and therefore it will fall back on the negotiation method to prevent unsafe behavior. The consensus based platooning method for local collision avoidance used in [16] is unusual in the AGV domain. That work makes no claims about completeness, but does consider the trivial consensus where all vehicles stop in a deadlock. In [21], a recent system for conflict avoidance based on time headway is shown to significantly reduce intersection crossing time and allow more vehicles to operate in the same floor-space compared to a traditional reservation based strategy.

It is shown, in [4], that platooning provides superior throughput to the earlier reservation based systems, and that if a solution exists it is optimal, but not that a solution exists on all roadmaps. More importantly a set of conditions, which must hold for a solution to exist, is not given. The consensus algorithm in [16] also shows improved throughput in concert with a scheduling approach, but does not prove convergence. An example of a resolution complete algorithm based on spatial reservation is [9]. Neither per-intersection optimal platooning nor per-vehicle consensus have been proven complete. The lack of guarantees is an important limitation of platooning methods for collision avoidance. The research gap identified is the lack of investigation into the range of motion conflict situations that can be resolved with platooning methods.

2. METHOD

Conflict resolution based on platooning, with speed choice by an intersection controller was implemented with a vehicle to intersection messaging scheme. The full site is divided into zones, each one containing a single intersection. Each AGV in the fleet has a copy of the roadmap which is static. The fleet controller interfaces with the warehouse management system to get the next material transfer job, consisting of a pick location and a drop location. All jobs are assumed to be of unit size and each AGV has a capacity of one unit. With these assumptions, a straightforward policy is to assign the next job to the AGV nearest to the pick location - first-come-first-served scheduling. When an AGV receives a new job, it finds the shortest path through the roadmap using the Floyd-Warshall algorithm. Next it must send its planned path to the intersection controller for the zone it currently occupies. The intersection controller stores the plan and current position of every AGV approaching the conflict point of the intersection. Every time it receives a new plan it must recalculate the approach speed for every approaching AGV to minimize total travel time without collision. This will happen every time an AGV enters the zone from somewhere else, or an AGV within the zone is assigned a new job.

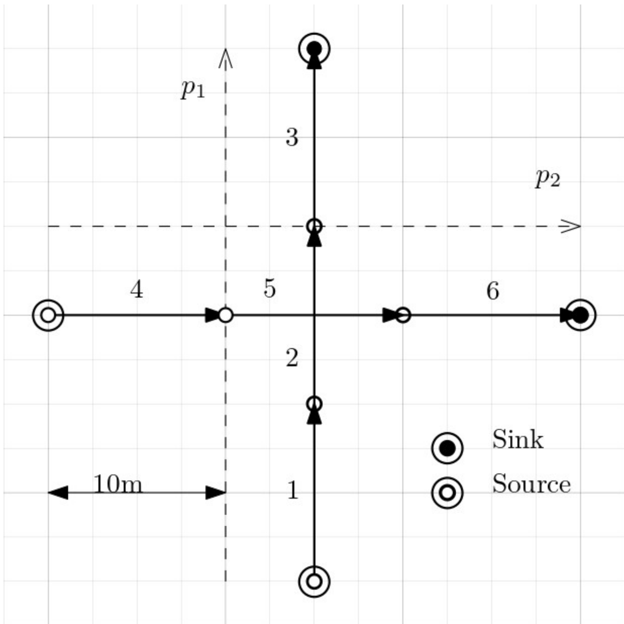

Figure 1. Intersection layout

The Floyd-Warshall algorithm holds some advantages in execution time and simplicity over commonly used Dijkstra and A* algorithms for finding the shortest path through a directed acyclic graph with positive edge weights. The roadmap can be expressed as such a graph with the edge weight corresponding to the distance. To exploit the static nature of the roadmap it is desirable to pre-calculate as much as possible, for fast run-time performance. Floyd-Warshall proceeds by exhaustive search over all edges in $O(N^3)$ operations using $O(N^3)$ memory in N the number of edges [6]. For each node, the next node in the shortest path to every destination is recorded. The result is N^2 shortest path trees linking every origin-destination pair, each of which can be traversed in $O(P)$ operations at run-time where P is the length of the path. Traversing a stored tree at run-time will be faster than graph search, even using efficient heuristic methods such as A*. The maximum number of nodes in the graph will be limited by the memory available for pre-computation, but memory is cheap and even a basic server should be sufficient for the largest sites with the most intricate networks.

The intersection controller was implemented based on [4]. The surrounding lanes are first discretized into segments. The intersection shown in Figure 1 is divided into six segments, each of length 10 meters. The critical segments are the two that cross in the center. There are two routes defined, one starting on the left and traveling to the right and the other starting at the bottom and traveling up. One AGV takes route 1 and the other takes route 2. If they both travel at maximum speed they will collide in the center.

The dynamic model for each AGV assumes they are able to exactly follow the path, and attempt to reach the target speed for each segment subject to a limited rate of acceleration of a m/s^2.

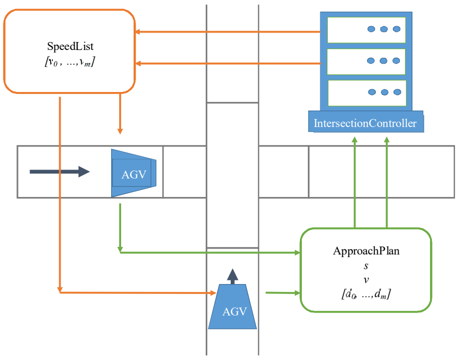

Figure 2. Messages exchanged by participants approaching intersection.

The ApproachPlan message sent by the AGV contains a sequence of segments, which it intends to traverse, along with its current distance along the first one. The flow of messages is shown in Figure 2. The SpeedList sent by the intersection controller contains the optimal speed for every segment in the plan. The speeds can be found with the nonlinear program in Equation 1.

$$\min_{\boldsymbol{\phi}} J_T = \boldsymbol{d}^T\boldsymbol{\phi} \tag{1}$$

subject to

$$\boldsymbol{f}(\boldsymbol{\phi}) \leq 0$$

$$\boldsymbol{\phi}_{min} \leq \boldsymbol{\phi} \leq \boldsymbol{\phi}_{max}$$

Each parameter is the inverse of the velocity of that segment, for one AGV:

$$\phi_{i,m} = \frac{1}{v_{i,m}} \tag{2}$$

Over all the known plans they are assembled into a vector

$$\boldsymbol{\phi} = [\phi_{1,1}, \dots, \phi_{1,M_1}, \cdots, \phi_{N,M_N}] \tag{3}$$

Where N is the number of AGV on approach and M_i is the number of segments in the plan of the ith AGV.

The nonlinear constraint $\boldsymbol{f}(\boldsymbol{\phi})$ enforces the separation between each pair based on the entry and exit time to the conflict segment, which depends on the speed at each preceding segment. Only elements with $j > i$ are included as the matrix A is symmetric and the diagonal elements are not useful.

$$f(\boldsymbol{\phi}) = [A_{1,2}, \dots, A_{i,j}, \dots, A_{N-1,N}]\ \forall i \in N, j > i \tag{4}$$

The constraint between each pair of AGVs can be expressed as in Equation 5.

$$A_{i,j} = (\beta_{i,j} + \beta_{i,j})^2 - (\alpha_{i,j} - \alpha_{j,i})^2 \tag{5}$$

$\alpha_{i,j}$ and $\beta_{i,j}$ have the units of time and are related to the arrival time $\omega_{i,j}^{min}$ and departure time $\omega_{i,j}^{max}$ at the conflicted segment by Equations 6 and 7 .

$$\alpha_{i,j} = \omega_{i,j}^{max} + \omega_{i,j}^{min} \tag{6}$$

$$\beta_{i,j} = \omega_{i,j}^{max} - \omega_{i,j}^{min} \tag{7}$$

Similarly

$$\alpha_{j,i} = \omega_{j,i}^{max} + \omega_{j,i}^{min}$$

$$\beta_{j,i} = \omega_{j,i}^{max} - \omega_{j,i}^{min}$$

The simplification of constant speed over each segment means that the arrival and departure time are related to $\boldsymbol{\phi}$ and the length of each segment $d_{i,j}$ by Equations 8 and 9:

$$\omega_{i,j}^{min} = \sum_{i=1}^{p_i} d_{i,j}\phi_{i,j} \tag{8}$$

$$\omega_{i,j}^{max} = \omega_{i,j}^{min} + \sum_{i=1}^{q_i} d_{i,j}\phi_{i,j} \tag{9}$$

where the total number of segments traversed by each vehicle M_i has been split into p_i segments approaching the conflict and q_i conflicted segments such that

$$M_i = p_i + q_i$$

PRELIMINARY RESULTS

Using the simple roadmap shown in Figure 1 with only two routes some failure cases can be examined. One set of failures is possible due to the start positions of the AGV. As all of the speeds must be recomputed every time a new AGV arrives or gets a new job, a valid plan may be requested for any starting location along the two zones. The only constraint considered in the optimization is the minimum and maximum speed of each segment.

The messaging protocol and dynamic equation were implemented in Python and Equation 1 solved using 'trust-constr' method [3] within scipy.optimize [1]. This method can benefit from analytical constraints if they are available. With the roadmap containing six segments only, only four need to be included in $\boldsymbol{\phi}$ because the speed on segments after the intersection can be chosen freely. For two AGVs traversing two segments each before leaving the conflict zone the optimization vector is $\boldsymbol{\phi} \in \mathbb{R}^4$. From Equation 4 there is only one constraint and $\boldsymbol{f}(\boldsymbol{\phi})$ has only one element as there is only one unique pair.

To provide a basic test case, both vehicles begin at the start of their approach lane, 10m from the conflict. The conflicted segment in both paths is 10 m in length. The maximum speed is 10 m/s, corresponding to $\phi_{min} = 0.1$. Convergence is attained in 94 function evaluations. The total travel time objective at the minimum is 5 seconds. The result $\boldsymbol{\phi} = [0.2, 0.1, 0.1, 0.1]$ indicates one vehicle slows down to 5m/s for the first segment to allow the other to pass before it continues at full speed.

The next test has both vehicles at the start of the conflict segment. There is no solution that can prevent both vehicles occupying the conflict as they both start there but the behavior of the solver is informative. After 829 function evaluations the termination condition appeared to be satisfied, but in fact the lower bound on $\boldsymbol{\phi}$ was not respected by the result. The speeds for one vehicle are impossibly high. This suggests that the appearance of multiple vehicles inside the conflict zone should be ruled out by the intersection controller software before attempting to solve for the optimal speeds.

Another test case has one vehicle positioned immediately before the conflict zone. The second is already crossing. This could occur if the first vehicle had stopped to complete a job at this position, and triggered the re-plan by requesting a set of speeds for the route to its next job. From this position, traveling at the minimum speed, it would enter the conflict zone before the departure of the crossing vehicle: as a result there is no feasible solution to Equation 1. This could cause convergence to fail much like the previous case but could occur in normal operation.

The final test case has each vehicle part way along the approach lane, stationary. As the acceleration limit is not taken into account in the optimization, the target speeds provided by the intersection controller may not be attained before arrival at the conflict zone. As both have the same acceleration rate they will arrive at the same time and collide.

3. FURTHER WORK

The tests conducted demonstrate that if platooning alone is used to coordinate the motion of a number of AGVs, workarounds are required for certain situations where one AGV needs to come to a complete stop or change route. The next step is to implement some of these workarounds and describe the types of roadmap where they need to be used often.

The constraints described here can be expressed in quadratic form allowing the analytical Hessian to be provided to the solver. Doing this may improve convergence with the 'trust-constr' solver. Another improvement to the described intersection controller would be the addition of constraints between the speeds in adjacent segments to reflect the limited acceleration of the AGV. Going further in this direction, the effect on convergence of solving for the optimal acceleration in each segment might be investigated.

A comparison in terms of both existence and quality of solutions with a consensus algorithm would reveal whether the limitations of platooning identified are fundamental or due to the harsh discretization required for optimality.

A simulation of multiple intersections within a whole site would reveal whether maximum throughput at each intersection leads to reduced trip times across multiple intersections, or whether this rule should be modified to account for downstream traffic

4. ACKNOWLEDGMENTS

This research was made possible thanks to the financial support of a full-time EPSRC Doctoral Training Partnership Studentship - Institute for Transport Studies and also thanks to the financial support of CASE partner Guidance Automation Limited.

5. REFERENCES

[1] 2019. Optimization and Root Finding (scipy.optimize) minimize(method='trust-constr'). https://docs.scipy.org/doc/scipy/reference/optimize.minimize-trustconstr.html#optimizeminimize-trustconstr

[2] Lubomír Bakule. 2008. Decentralized control: An overview. *Annual Reviews in Control* 32, 1 (2008), 87–98. https://doi.org/10.1016/j.arcontrol. 2008.03.004

[3] Andrew R Conn, Nicholas IM Gould, and Ph L Toint. 2000. *Trust region methods*. Vol. 1. Siam.

[4] Valerio Digani, M. Ani Hsieh, Lorenzo Sabattini, and Cristian Secchi. 2019. Coordination of multiple AGVs: a quadratic optimization method. *Autonomous Robots* 43, 3 (2019), 539–555. https://doi.org/10.1007/ s10514-018-9730-9

[5] Valerio Digani, Lorenzo Sabattini, Cristian Secchi, and Cesare Fantuzzi. 2014. Hierarchical traffic control for partially decentralized coordination of multi AGV systems in industrial environments. *Proceedings-IEEE International Conference on Robotics and Automation* (2014), 6144–6149. https://doi.org/10.1109/ICRA.2014.6907764

[6] Michael Alexander Djojo and Kanisius Karyono. 2013. Computational load analysis of Dijkstra, A*, and Floyd-Warshall algorithms in mesh network. *Proceedings of 2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems, ROBIONETICS 2013*(2013), 104–108. https://doi.org/10.1109/ROBIONETICS.2013.6743587

[7] Mariagrazia Dotoli, Alexander Fay, Marek Miśkowicz, and Carla Seatzu. 2019. An overview of current technologies and emerging trends in factory automation. *International Journal of Production Research* 57, 15-16 (2019), 5047-067. https://doi.org/10.1080/00207543.2018.1510558

[8] Ivica Draganjac, Damjan Miklic, Zdenko Kovacic, Goran Vasiljevic, and Stjepan Bogdan. 2016. Decentralized Control of Multi-AGV Systems in Autonomous Warehousing Applications. *IEEE Transactions on Automation Science and Engineering* 13, 4 (2016), 1433–1447. https://doi.org/10.1109/TASE.2016.2603781

[9] Ivica Draganjac, Tamara Petrović, Damjan Miklić, Zdenko Kovačić, and Juraj Oršulić. 2020. Highly-scalable traffic management of autonomous industrial transportation systems. *Robotics and Computer-Integrated Manufacturing* 63, July 2018 (2020), 101915. https://doi.org/10.1016/j. rcim.2019.101915

[10] Kurt Dresner. 2008. A Multiagent Approach to Autonomous Intersection Management. *Journal of Artificial Intelligence Research* 31 (2008), 591–656.

[11] Bai Li, Hong Liu, Duo Xiao, Guizhen Yu, and Youmin Zhang. 2017. Centralized and optimal motion planning for large-scale AGV systems: A generic approach. *Advances in Engineering Software* 106 (2017), 33–46. https://doi.org/10.1016/j.advengsoft.2017.01.002

[12] Pablo Alvarez Lopez and Jakob Erdmann. 2016. Supporting additional elements in a simulation using Netedit. In *SUMO User Conference 2016*. https://elib.dlr.de/124399/

[13] Toshiyuki Miyamoto and Kensuke Inoue. 2016. Local and random searches for dispatch and conflict-free routing problem of capacitated AGV systems. *Computers and Industrial Engineering* 91 (2016), 1–9. https://doi.org/10.1016/j.cie.2015.10.017

[14] Jackeline Rios-Torres and Andreas A. Malikopoulos. 2017. A Survey on the Coordination of Connected and Automated Vehicles at Intersections and Merging at Highway On-

Ramps. *IEEE Transactions on Intelligent Transportation Systems* 18, 5 (2017), 1066–1077. https://doi.org/10.1109/TITS.2016.2600504

[15] Lorenzo Sabattini, Mika Aikio, Patric Beinschob, Markus Boehning, Elena Cardarelli, Valerio Digani, Annette Krengel, Massimiliano Magnani, Szilard Mandici, Fabio Oleari, Christoph Reinke, Davide Ronzoni, Christian Stimming, Robert Varga, Andrei Vatavu, Sergi Castells Lopez, Cesare Fantuzzi, Aki Mayra, Sergiu Nedevschi, Cristian Secchi, and Kay Fuerstenberg. 2018. The PAN-robots project: Advanced automated guided vehicle systems for industrial logistics. *IEEE Robotics and Automation Magazine* 25, 1 (2018), 55–64. https://doi.org/10.1109/MRA.2017.2700325

[16] Kumiko Tadano and Yoshiharu Maeno. 2019. Scalable control system for dense transfer vehicles in flexible manufacturing. *2019 IEEE Conference on Control Technology and Applications (CCTA)* 3 (2019), 325–331.

[17] Jur P. Van Den Berg and Mark H. Overmars. 2005. Prioritized motion planning for multiple robots. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS* (2005), 2217–2222. https://doi.org/10.1109/IROS.2005.1545306

[18] Iris F.A. Vis. 2006. Survey of research in the design and control of automated guided vehicle systems. *European Journal of Operational Research* 170, 3 (2006), 677-709. https://doi.org/10.1016/j.ejor.2004.09.020

[19] Jingjin Yu and Steven M. Lavalle. 2013. Planning optimal paths for multiple robots on graphs. *Proceedings - IEEE International Conference on Robotics and Automation* (2013),3612–3617. https://doi.org/10.1109/ICRA.2013.6631084

[20] Jingjin Yu and Daniela Rus. 2018. An Effective Algorithmic Framework for Near Optimal Multi-robot Path Planning. (2018), 495–511. https: //doi.org/10.1007/978-3-319-51532-8_30 arXiv:1505.00200

[21] Zheng Zhang, Qing Guo, Juan Chen, and Peijiang Yuan. 2018. Collision-Free Route Planning for Multiple AGVs in an Automated Warehouse Based on Collision Classification. *IEEE Access* 6 (2018), 26022–26035. https://doi.org/10.1109/ACCESS.2018.2819199