Proceedings Paper:

Lambert, ED orcid.org/0000-0002-2297-0441, Romano, R orcid.org/0000-0002-2132-4077 and Watling, D orcid.org/0000-0002-6193-9121 (Accepted: 2020) Intersection Platooning for Distributed Conflict Resolution of an AGV Fleet. In: UNSPECIFIED 2020 IEEE 16th International Conference on Automation Science and Engineering, 20-21 Aug 2020, Online. . (Unpublished)

Intersection Platooning for Distributed Conflict Resolution of an AGV Fleet

Edward Derek Lambert[1] and Richard Romano[2] and David Watling[3]

***Abstract*— A proposal to use a series of connected intersection managers to achieve distributed conflict resolution for a fleet of AGV is examined based on the completeness and optimality of the optimization used to select the speeds. Quadratic constraints resulting from a simplified control model are shown to be non-convex by finding local minima on a small example problem. Local minima reduce throughput for cross traffic and cause collisions for AGVs in the same lane. An alternative constraints formulations is developed which results in a linear program to addresses this problem on a small example with simplified dynamics.**

I. 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 [1] and [2]. A crucial sub-problem is conflict resolution between multiple AGVs, without control of task assignment or scheduling.

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 [3]. Centralized optimization approaches improve on early ideas like First-Come-First-Served spatial reservation from [4] 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 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. Notably the conditions for solutions to exist which minimize energy consumption are given in [5].

It is shown, in [6], 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 [7] 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 [8]. 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.

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.

Edward Derek Lambert and Richard Romano and David Watling are with the Institute for Transport Studies, University of Leeds, 34-41 University Road, Leeds, LS2 9JT, UK

[1]tsedl@leeds.ac.uk

[2]R.Romano@leeds.ac.uk

[3]D.P.Watling@its.leeds.ac.uk

II. METHOD

Platooning with speed choice by a centralized 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 [9]. 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.

The intersection controller was implemented based on [6]. 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.

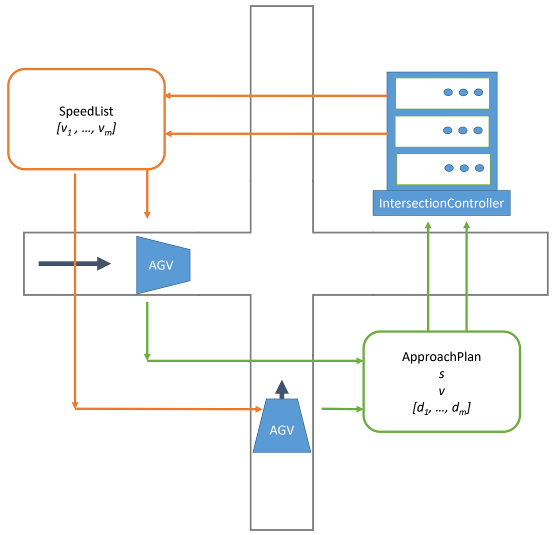

Fig. 1. Messages exchanged by participants approaching intersection.

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 1. 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 5.

A. Intersection Controller Objective

The objective is to minimize J_T the total travel time for all vehicles. It is convenient for exposition to optimize over the inverse of speed of each segment $\phi_k = 1/v_k$. Vehicle i submits a plan containing m_i segments before the conflict and n_i segments in conflict. The control model is based on the average speed of each approaching AGV over each segment. This is to simplify the description of the intersection controller, and assist analysis. More sophisticated motion models could take the place of Equation 6 and Equation 8 to create a similar type of problem with a convex travel time objective and non-convex constraints. The parameters for one vehicle can be collected in the vector $\boldsymbol{\phi}_i$ as shown in Equation 1

$$\boldsymbol{\phi}_i = [\phi_1, ..., \phi_{(m_i+n_i)}]^T \tag{1}$$

The parameters for p vehicles each traversing $(m_i + n_i)$ segments are assembled into a vector as in Equation 2

$$\boldsymbol{\phi} = [\boldsymbol{\phi}_\mathbf{1}^T, ..., \boldsymbol{\phi}_p^T]^T \tag{2}$$

Similarly, the length of each segment in plan i can be arranged into a vector

$$\boldsymbol{d}_i = [d_1, ..., d_{(m_i+n_i)}]^T \tag{3}$$

and collected for p vehicles into a vector as in Equation 4.

$$\boldsymbol{d} = [\boldsymbol{d}_\mathbf{1}^T, ..., \boldsymbol{d}_p^T]^T \tag{4}$$

This leads to the minimum travel time objective in Equation 5.

$$\begin{gathered} \min_{\boldsymbol{\phi}} \boldsymbol{J_T} = \boldsymbol{d}^T \boldsymbol{\phi} \\ \text{subject to} \\ \boldsymbol{\phi}_{max} > \boldsymbol{\phi} > \boldsymbol{\phi}_{min} \\ \boldsymbol{\phi}^T \boldsymbol{H}_{i,j} \boldsymbol{\phi} > \mathbf{0} \quad \forall i, j \in [1, p] \quad \text{with } j > i \end{gathered} \tag{5}$$

The condition $j > i$ in Equation 5 indicates that the number of constraints varies with the number of vehicles p as $\frac{p(p-1)}{2}$. This corresponds to one constraint between each pair of approaching AGVs.

B. Intersection Controller Timing Constraints

By definition, each intersection has a single conflict zone, the union of all segments which intersect there. This makes it possible to express the constraint that vehicles do not collide in terms of time. Vehicle i arrives at the first conflicted segment ω_i^{min} and departs from the last at ω_i^{max}. The following three subsections set out three alternative ways of expressing the collision avoidance constraints which have been evaluated. The arrival time is given by Equation 6. Considering average speeds, the departure time ω_i^{max} is also linear, this is given by Equation 8.

$$\omega_i^{min} = \sum_{k=1}^{m_i} \boldsymbol{d}_i[k] \boldsymbol{\phi}_i[k] = \boldsymbol{e}^T \boldsymbol{\phi}_i \tag{6}$$

where

$$\boldsymbol{e}[k] = \begin{cases} \boldsymbol{d_i}[k] & \forall k < m_i \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

and m_i is the number of segments on the path of vehicle i before arrival at the conflicted segment.

$$\omega_i^{max} = \omega_i^{min} + \sum_{i=1}^{n_i} \boldsymbol{d}_i[k] \boldsymbol{\phi}_i[k] = \boldsymbol{f}^T \boldsymbol{\phi}_i \tag{8}$$

where

$$\boldsymbol{f}[k] = \begin{cases} \boldsymbol{d_i}[k] & \forall k < m_i + n_i \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

and n_i is the number of segments on the path of vehicle i which are conflicted. Note that Equations 6 and 8 only depend on the $\boldsymbol{\phi}_i$ of vehicle i.

The timing constriant between each pair of vehicles can be expressed with a modulus operator as in Equation10.

$$|\alpha_i - \alpha_j| > \beta_i + \beta_j \tag{10}$$

Here

$$\alpha_i = \omega_i^{max} + \omega_i^{min} \tag{11}$$

represents the midpoint of the time vehicle i occupies the conflicted segment and

$$\beta_i = \omega_i^{max} - \omega_i^{min} \tag{12}$$

represents the range of the time either side of the midpoint, both scaled by a factor of two.

In matrix form

$$\alpha_i = \boldsymbol{f}^T \boldsymbol{\phi}_i + \boldsymbol{e}^T \boldsymbol{\phi}_i = \mathbf{1}_i^T \boldsymbol{A} \phi_i \tag{13}$$

with $\boldsymbol{A} = diag(\boldsymbol{f} + \boldsymbol{e})$

$$\beta_i = \boldsymbol{f}^T \boldsymbol{\phi}_i - \boldsymbol{e}^T \boldsymbol{\phi}_i = \mathbf{1}_i^T \boldsymbol{B} \phi_i \tag{14}$$

with $\boldsymbol{B} = diag(\boldsymbol{f} - \boldsymbol{e})$

The resulting linear program (with parameters $\in \mathbb{R}$) has $p-1$ constraints as each AGV is only constrained by the preceeding one.

1) Quadratic Constraints: Another way to treat the modulus operator in Equation 10, without forcing any particular arrival order is to square both sides as to give the expression in Equation 15.

$$\alpha_i^2 - \alpha_j^2 - 2\alpha_i\alpha_j - (\beta_i^2 + \beta_j^2 + 2\beta_i\beta_j) > 0 \tag{15}$$

Collecting terms by subscript gives

$$(\alpha_i^2 - \beta_i^2) - (\alpha_j^2 + \beta_j^2) - 2(\alpha_i\alpha_j + \beta_i\beta_j) > 0 \tag{16}$$

The matrix $\boldsymbol{\Lambda}_i j$ captures the constraints between a pair of vehicles and always contains four sub-matrices as shown in in Equation 17. It is compatible with $\boldsymbol{\phi}_{i,j} = [\boldsymbol{\phi}_i^T, \boldsymbol{\phi}_j^T]^T$, containing only the relevant speeds for vehicles i and j.

$$\boldsymbol{\Lambda}_{ij} = \begin{bmatrix} \boldsymbol{\Lambda}_{ij}^{ii} & \boldsymbol{\Lambda}_{ij}^{ij} \\ \boldsymbol{\Lambda}_{ij}^{ji} & \boldsymbol{\Lambda}_{ij}^{jj} \end{bmatrix} \tag{17}$$

Expanding

$$[\boldsymbol{\phi}_i^T, \boldsymbol{\phi}_j^T] \begin{bmatrix} \boldsymbol{\Lambda}_{ij}^{ii} & \boldsymbol{\Lambda}_{ij}^{ij} \\ \boldsymbol{\Lambda}_{ij}^{ji} & \boldsymbol{\Lambda}_{ij}^{jj} \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}_i \\ \boldsymbol{\phi}_j \end{bmatrix}$$
$$= \boldsymbol{\phi}_i^T \boldsymbol{\Lambda}_{ij}^{ii} \boldsymbol{\phi}_i + \boldsymbol{\phi}_j^T \boldsymbol{\Lambda}_{ij}^{jj} \boldsymbol{\phi}_j + \boldsymbol{\phi}_i^T \boldsymbol{\Lambda}_{ij}^{ij} \boldsymbol{\phi}_j + \boldsymbol{\phi}_j^T \boldsymbol{\Lambda}_{ij}^{ji} \boldsymbol{\phi}_i \tag{18}$$

makes it possible to compare terms with the scalar expression in Equation 16. This leads to the following expressions for the submatrices in Λ in terms of $\alpha_i = \mathbf{1}_T \boldsymbol{A}_i \boldsymbol{\phi}_i$ and $\beta_i = \mathbf{1}_T \boldsymbol{B}_i \boldsymbol{\phi}_i$

$$\boldsymbol{\Lambda}_{ij}^{ii} = (\boldsymbol{A}_i - \boldsymbol{B}_i)\mathbf{1}_i\mathbf{1}_i^T(\boldsymbol{A}_i - \boldsymbol{B}_i) \tag{19}$$

$$\boldsymbol{\Lambda}_{ij}^{jj} = -(\boldsymbol{A}_j + \boldsymbol{B}_j)\mathbf{1}_j\mathbf{1}_j^T(\boldsymbol{A}_J + \boldsymbol{B}_j) \tag{20}$$

$$\boldsymbol{\Lambda}_{ij}^{ij} + \boldsymbol{\Lambda}_{ij}^{ijT} = -2(\boldsymbol{A}_j + \boldsymbol{B}_j)\mathbf{1}_j\mathbf{1}_i^T(\boldsymbol{A}_i + \boldsymbol{B}_i) \tag{21}$$

For more than two vehicles this can be arranged into a block diagonal matrix $\boldsymbol{H}_{ij}$ which is compatible with the input parameters, but still only represents the constraints between a pair.

Expressed in this way it is clear the constraints are quadratic and it is trivial to differentiate twice to find the Hessian is the stack of constraint matrices $[\boldsymbol{H}_{ij}, \ldots]$. The objective is certainly convex as it is linear but the constraints may not be. If the Hessian of the constraints is positive semi-definite then they are convex and interior point methods will either find the global optimum or prove that there is no feasible solution [10]. The Hessian depends on the parameters of the roadmap, the number of approaching vehicles and their distance from the conflict.

2) Linear FIFO Constraints: If the order in which the AGV cross the conflict zone is fixed to be First-In-First-Out, the timing constraint is linear. The parameters for each AGVmust first be sorted in order of distance from the conflict zone. Then there is one constraint between each adjacent pair so $p-1$ constraints total for p vehicles. These can be expressed in the form $\boldsymbol{A}_{ub}\phi \leq \boldsymbol{b}_{ub}$ as in Equation 22. This is correct for two AGVs arranged in distance order, each traversing one approach and one conflict segment.

$$\begin{bmatrix} -d_1 & 0 & d_3 & d_4 \\ \vdots & & & \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} \leq \begin{bmatrix} 0 \\ \vdots \end{bmatrix} \tag{22}$$

III. Results

The simulated setup is shown in Figure 2, with two AGVs approaching the crossroads one from each source node. Each vehicle is stationary at the start of its respective lane at t=0. Both vehicles request speed guidance for three segments, taking them directly across the intersection. First the Hessian is examined, then the results of a 10 second simulation of vehicles with limited acceleration are reported.

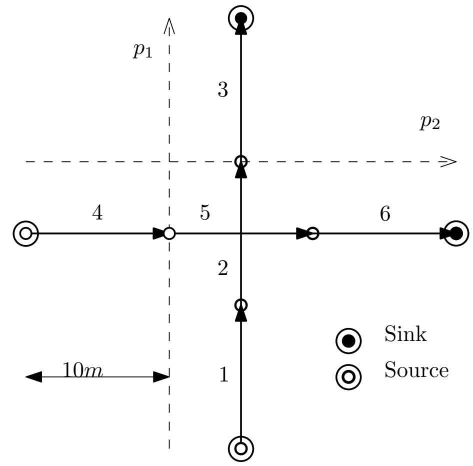

Fig. 2. Intersection layout with two conflicting routes.

The Hessian $\boldsymbol{H}_{ij}$ was evaluated for the simple crossroads shown in Figure 2 with two approaching AGVs. In this case, only $\boldsymbol{H}_{01}$ is included and this is identical to pairwise $\Lambda_{01} \in R^{(4\times4)}$ given by Equation 17 as there are only two vehicles in total. This has eigenvalues $\boldsymbol{e_1} = [0, 2000, -400, 0]$, so the convexity of the constraints is not proven. The linear objective is convex by definition but cannot be strictly convex, as strict convexity precludes linear regions. Convexity permits multiple minima but ensures that every local minimum is a global minimum.

Using the ‘trust-constr’ solver in scipy.optimize and providing the derived Jacobian and Hessian of both cost and constraints, the execution time to find the optimal speeds was 0.22 seconds. The minimum was found to be $J_T = 5$s with parameters $\boldsymbol{\phi_1} = [0.20000006, 0.10000002]$ and $\boldsymbol{\phi_2} = [0.10000001, 0.10000001]$. This is close to the true value of

$\phi_1 = [0.2, 0.1]$ and $\phi_2 = [0.1, 0.1]$ and more precision can be achieved by tuning the value of $'gtol'$. Smaller tolerance values get closer to the true min. A value of $'gtol' = 1e-14$ was used, leading to constraint error of 2.4707×10^{-06}. There is another equally valid minimum with $J_T = 5$s which is not found with the initial guess in which all parameters were set to ϕ_{min}.

If the scenario is modified so the second vehicle starts 1m closer to the intersection, both minima are no longer equally costly. Now the global minimum where the vehicle at the start slows down to 5.26m/s to allow the closer vehicle to pass in front of it leads to $J_T = 4.8$s. The alternative order where the AGV 9m away slows to 4.5m/s leads to $J_T = 5.22$s. The Hessian was evaluated and the eigenvalues found to be $\boldsymbol{e_2} = [0, 1848, -400, 0]$. They do not all have the same sign, so the convexity of the constraints is not proven. The 'trust-constr' solver, provided with analytical Jacobian and Hessian converged to either minimum depending on the initial guess. As a result the speed choice for larger numbers of vehicles could be sub-optimal if no steps are taken to explore the cost surface such as trying multiple initial guesses for each problem.

Another test involved vehicles approaching in the same lane. Only one AGV may occupy the conflict zone at a time according to the constraints so each additional AGV should slow down enough for the preceding one to have left the conflict zone by the time it arrives. At the global minimum for the two vehicle example, traffic from conflicting directions should be interleaved on a FIFO basis. However, the sub-optimal local minima lead to a more serious problem here because vehicles in a queue will collide if those further back are given higher speeds. A workaround based on 'car-following' behavior might be implemented at an individual vehicle level, based on the distance to the vehicle in front. However individual behavior contradicting the speed instructions from the intersection manager may lead to collisions with conflicting traffic.

The linear program described in Section II-B.2 has fewer constraints and is a convex problem. It was solved with 'linprog' solver in scipy.optimize for the same scenario. This was much faster and guaranteed to find the minimum (with FIFO arrival) or return an error because the problem is infeasible. These are attractive properties for a real time controller. No cases were found where the FIFO minimum was worse than the local minimum found with flexible ordering, but these are expected to exist.

Simulating vehicles with a limited constant rate of acceleration of 1m/s^2 with these target speeds led to a collision at $t = 5.3$s. This is because the approach lane length of $d = 10$m is too short for either vehicle to reach their target of 5m/s and 10m/s respectively. As they have the same acceleration rate, their speed profile is identical and they collide 0.69m away from the center of the intersection. They were assumed to have a radius of 0.5m. If the acceleration rate is increased to 5m/s^2 there is no collision. The minimum acceleration rate for successful avoidance was found to be 2m/s.

IV. CONCLUSIONS

The existence of sub-optimal local minima in the solution to Equation 5 is an important limitation if intersection management alone is to be used to solve motion coordination across a site. The intention was to create car-following behavior by applying the same rules to traffic in the same lane. At the global minimum for the two vehicle example this is possible because the order of approach will be preserved. At one of the local minima the order of arrival may change, indicating in lane overtaking which may not be possible. Applying a supervisory system which affected the speeds could lead to collision with cross traffic.

This suggests the constraints should be modified to enforce the order of arrival. For AGVs in the same lane, the order of arrival is a hidden constraint which leads to infeasible solutions. If the constraints with FIFO ordering are used for cross traffic as well, the problem is a linear program, which is convenient to solve. The downside is the loss of ordering flexibility in the solution. Therefore, the restatement as a mixed integer linear program could be beneficial. An integer solver could explicitly evaluate all the possible arrival orders and return the global optimum. Depending on the exact formulation and the number of AGV per intersection at any one time, this could be a feasible solution for a real-time intersection controller.

REFERENCES

[1] I. F. Vis, "Survey of research in the design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 170, no. 3, pp. 677–709, 2006.

[2] M. Dotoli, A. Fay, M. Miśkowicz, and C. Seatzu, "An overview of current technologies and emerging trends in factory automation," *International Journal of Production Research*, vol. 57, no. 15-16, pp. 5047–5067, 2019. [Online]. Available: https://doi.org/10.1080/00207543.2018.1510558

[3] J. Rios-Torres and A. A. Malikopoulos, "A Survey on the Coordination of Connected and Automated Vehicles at Intersections and Merging at Highway On-Ramps," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1066–1077, 2017.

[4] K. Dresner, "A Multiagent Approach to Autonomous Intersection Management," *Journal of Artificial Intelligence Research*, vol. 31, pp. 591–656, 2008.

[5] A. A. Malikopoulos, C. G. Cassandras, and Y. J. Zhang, "A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections," *Automatica*, vol. 93, pp. 244–256, 2018. [Online]. Available: https://doi.org/10.1016/j.automatica.2018.03.056

[6] V. Digani, M. A. Hsieh, L. Sabattini, and C. Secchi, "Coordination of multiple AGVs: a quadratic optimization method," *Autonomous Robots*, vol. 43, no. 3, pp. 539–555, 2019. [Online]. Available: https://doi.org/10.1007/s10514-018-9730-9

[7] K. Tadano and Y. Maeno, "Scalable control system for dense transfer vehicles in flexible manufacturing," *2019 IEEE Conference on Control Technology and Applications (CCTA)*, no. 3, pp. 325–331, 2019.

[8] I. Draganjac, T. Petrović, D. Miklić, Z. Kovačić, and J. Oršulić, "Highly-scalable traffic management of autonomous industrial transportation systems," *Robotics and Computer-Integrated Manufacturing*, vol. 63, no. July 2018, p. 101915, 2020. [Online]. Available: https://doi.org/10.1016/j.rcim.2019.101915

[9] M. A. Djojo and K. Karyono, "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*, pp. 104–108, 2013.

[10] S. Boyd and L. Vandenberghe, *Chapter 3: Convex Functions*, 7th ed. Cambridge University Press, 2004. [Online]. Available: www.cambridge.org/9780521833783