STARC: Low-power Decentralized Coordination Primitive for Vehicular Ad-hoc Networks

Patrick Rathje[1], Valentin Poirot[1,2], Olaf Landsiedel[1,2]
[1] Kiel University, Kiel, Germany
[2] Chalmers University of Technology, Göteborg, Sweden
mail@patrickrathje.de, {vpo;ol}@informatik.uni-kiel.de

***Abstract*—Intersections are the bottlenecks of road networks. Coordination mechanisms for intersection crossing greatly affect the efficiency of road utilization. Typically, coordination is done by implanting local infrastructure, whether signs, traffic lights, or through common, well known-rules shared by all users.**

In this paper, we introduce STARC, a decentralized intersection management protocol for future connected vehicles and other traffic participants. With STARC, all participants coordinate their movement using reservations to guarantee safe crossings. To enable cost-efficient deployment, STARC does neither rely on any centralized infrastructure, such as traffic lights, nor centralized wireless intersection coordinators, like virtual traffic lights. STARC targets small, cheap, and energy-efficient platforms and the open low-power wireless standard 802.15.4 so that all participants in road traffic could take advantage of it, including vehicles, bikes, electric scooters, and even pedestrians. STARC builds on low-power wireless communication with A^2-Synchrotron and multi-hop routing as a communication substrate and provides distributed transaction, election, and handover mechanisms to manage the intersection cooperatively. We show that STARC reduces average waiting times by up to 50% compared to a fixed traffic light schedule in traffic volumes with less than 1000 vehicles per hour. Moreover, we illustrate a platoon extension that allows STARC to outperform traffic lights even at traffic loads of over 1000 vehicles per hour.

***Index Terms*—cooperative intersection management, intelligent transportation systems, synchronous transmissions, autonomous driving, vehicle-to-vehicle communication, low-power vehicular ad-hoc networks**

I. Introduction

Intelligent Transportation Systems (ITS) lay the foundations for new, efficient ways of mobility. By adopting ITS, traffic delay, fuel consumption, and greenhouse gas emissions could be reduced [1, 2]. At the same time, vehicle-to-everything (V2X) communication could replace infrastructures like traffic signs and lights and minimize infrastructure costs [1, 3]. Protocols in this domain often rely on the presence of cellular network coverage or central servers for coordination [1, 2]. In contrast, decentralized approaches do not require additional infrastructure as they solely rely on communication between the vehicles [4, 5]. However, most of these focus on bandwidth intense technologies such as 802.11p or 5G device-to-device (D2D) and are inherently limited to platforms with sufficient compute and energy resources. Moreover, many proposals neglect a detailed performance and reliability evaluation in realistic communication environments [6].

In contrast, we argue that there is a need for distributed, safe, and efficient protocols for traffic coordination able to run on resource-constrained platforms.

Challenges. Algorithms for robust and fault-tolerant coordination have an inherent complexity that defines a stark contrast to the resource-constrained low-power wireless platforms we target. Thus, we need to devise algorithms and a system design that can deal with unreliable wireless communication, complex routing, and group-membership in a delay-sensitive environment — without sacrificing efficiency and safety.

Approach. We introduce STARC, a decentralized reservation-based protocol that uses cheap, low-power wireless radios to enable energy-efficient vehicle-to-vehicle communication. We build our coordination protocol on top of A^2-Synchrotron (Synchrotron), a low-latency and energy-efficient communication primitive for all-to-all communication [7]. With STARC, traffic participants reserve lanes to cross the intersection. We provide transaction semantics, and all participants coordinate to commit on a shared access pattern jointly. As a result, vehicles have unique access to different parts, i.e., lanes, of the intersection, ensuring that at most, one car can use a given lane. Once a car has crossed the intersection, it releases its reservation and leaves the communication network, allowing the next car to continue its journey.

Our design and implementation for IEEE 802.15.4 radios allow STARC to operate on energy-restricted devices to support all road users, including cyclists and pedestrians.

Contributions. This paper contributes the following:

- We present STARC, a decentralized protocol for autonomous intersection management.
- STARC provides fault-tolerant, distributed coordination, and transactions in dynamic networks.
- Targeting all traffic participants, we design and implement STARC for resource-constrained IoT platforms.
- We evaluate STARC on a simulated intersection showing its efficiency and fault-tolerance under injected interference and radio failures.

Outline. After we cover background, related work and a introduction to Synchrotron in Section II, we present the design in Section III and its evaluation in Section IV. Finally, we conclude our work in Section V.

II. Background

Intelligent Transportation Systems build on computer systems to aid traffic optimization. Brake assistants, crash, and congestion warnings are example applications. Vehicles may also form platoons to save fuel and increase road utilization [8]. Such advanced intents require communication of the vehicles with each other (V2V) or with the infrastructure (V2I) [3]. Available protocols for V2V and V2I communication are based on IEEE 802.11p or cellular protocols (LTE or 5G) [9]. In Section II-A, we review current solutions for intersection coordination. Then, in Section II-B, we introduce our communication substrate, Synchrotron.

A. Related Work

Vanmiddlesworth et al. present a reservation coordination-protocol based on claims [4]. In their approach, cars broadcast claims for the intersection repeatedly, and the one with the most dominant claim may proceed. While they show that their approach doubles the throughput compared to ordinary stop signs in low volume scenarios, the protocol itself can not handle communication failures safely [10].

Virtual Traffic Lights (VTL) replace physical, centralized traffic lights with a virtual coordination infrastructure by sending messages between the participants [5]. In case of potential conflicting directions, vehicles elect a single vehicle as the coordinator, which generates and broadcasts a schedule. As soon as the coordinator is allowed to drive, it will handover coordination to another vehicle. Ferreira et al. show that Virtual Traffic Lights can reduce CO_2 emission by up to 18% [11]. VTL are shown to improve the driving experience as they reduce mean travel and waiting time [12]. Further optimizations are made by Sommer et al. by dynamically changing the phase length based on the number of waiting cars [10].

The protocol of Hassan and Rakha reduces the number of messages sent by letting only the leading vehicles of each lane create the schedule [13]. Each leading vehicle then propagates the schedule back through the lanes. Naumann et al. propose a semaphore-based algorithm which only allows one vehicle to stay in each critical region of an intersection [14].

In contrast, STARC builds its crossing schedule via a 2-Phase Commit-like network-wide agreement. Since all participants must participate and agree for a schedule to be valid, STARC achieves fairness and safety: All participants eventually cross the intersection, and at most, one schedule can be selected at any instant. Moreover, we implement STARC for low-power wireless radios. Thus, any road user can take advantage of the protocol.

B. Synchronous Communication with Synchrotron

A^2-Synchrotron (Synchrotron) [7] is a wireless communication primitive combining all-to-all data sharing and in-network processing to enable interaction between all nodes in low-power wireless networks. Synchrotron provides multi-hop communication and inherently supports mobile environments. Moreover, it readily integrates channel-hopping and cryptographic primitives to ensure a robust protocol. At its heart, it makes use of two fundamental techniques: Synchronized transmissions and data aggregation.

Synchronized Transmissions. In Synchrotron, the nodes communicate in rounds. Each round is divided into multiple slots in which nodes try to receive a packet or transmit. Although multiple nodes might transmit concurrently during the same slot, some nodes can correctly receive and decode packets due to the capture effect [15]. Between two rounds, the radio is turned off to save energy.

Data Aggregation. Upon receiving a packet, a node processes the information using an application-specific merge-operator, marks its participation, and disseminates the result. This approach combines data collection, processing, and dissemination at once.

III. Design

Overview. The STARC protocol guarantees safe intersection crossing without the need for central infrastructure. We rely only on local, low-power wireless communication. We divide the protocol into two parts (1) Reservation-based movement coordination and (2) Dynamic network support; see Fig. 1a.

System Assumptions. We assume that all road users are equipped with IEEE 802.15.4 radios. Moreover, we assume the presence of position measurement devices, e.g., some sensors and a unique vehicle identifier (ID). Furthermore, we assume access to information about the intersections, i.e., lane-layout, size, and possible directions. All vehicles have (limited) computing capabilities and can plan a trajectory they want to perform. Another assumption is non-byzantine behavior: All participants are honest and follow the protocol - no vehicle purposely tries to disrupt it.

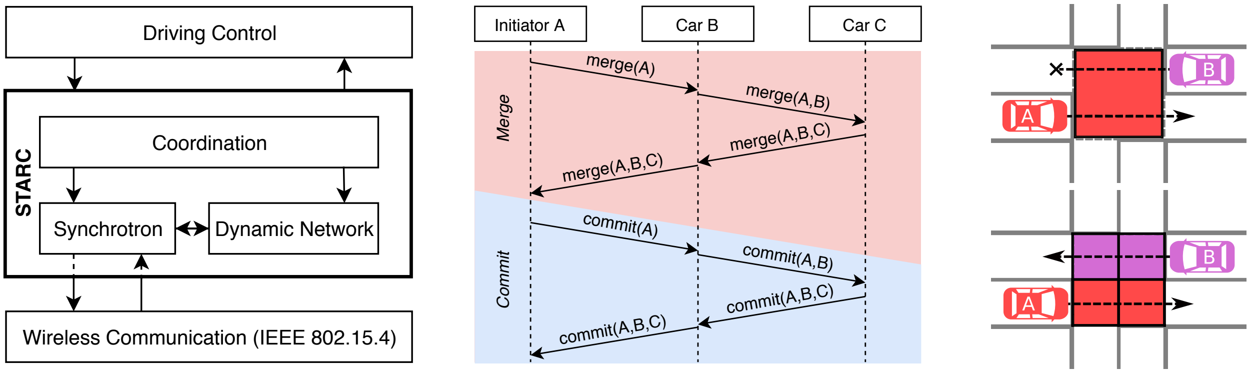
A. Distributed Reservation Coordination

Intersection setup. We divide the intersection into a $n \times n, n \in \mathbb{N}_+$ grid of even tiles [1]. Each tile covers a specific area and possesses a unique address. The tile-layout of the intersection is known to every vehicle.

Tile reservation. A vehicle requests a reservation for tiles based on the tile addresses. Besides the tiles, a request contains the unique identifier of the vehicle and its priority. We base the priority on the arrival time and enable fairness by priority [14]: A later arrival time results in a lower priority.

Once a vehicle desires to pass the intersection, it plans its path over the intersection and tries to reserve all the tiles on its way, see Fig. 1c. No notion of crossing time is used as we have to ensure safety also when vehicles take longer to pass the intersection as initially planned. The vehicle may only start when all needed tiles are reserved [14]. When the vehicle is leaving the intersection, it frees its reserved tiles, so that other vehicles may reserve them.

Our protocol builds on top of Synchrotron since it is designed explicitly for simultaneous data sharing and processing. We use this property to create a Synchrotron round



(a) Architecture Overview

(b) Coordination Round

(c) Granularity

Fig. 1. *(a)* The STARC protocol builds coordination on top of Synchrotron and adds support for dynamic networks. *(b)* The leader starts a coordination round in the merge phase. As soon as all vehicles marked their participation, the commit phase is started and the merged schedule established. *(c)* A granularity of 1 (top) supports only a single vehicle while a granularity of 2 (bottom) enables parallel crossings.

for parallelly collecting and merging the reservations. This coordination round is split into two phases, as depicted in Fig. 1b:

1. *Merge phase*: Individual reservation requests are collected and merged into a feasible schedule.
2. *Commit phase*: If all members participated, this phase disseminates the resulting intersection-wide schedule to ensure consensus.

Merge phase. The merge phase uses a custom merge-operator that simultaneously merges and solves occurring conflicts according to individual priorities. Additionally, the operator is order-invariant and creates partial schedules of the participated vehicles. The leader node starts the merge phase (and thus the coordination round) with the initial transmission. The progress is tracked using participation flags in the packet. With the participation of all members in the network, the schedule and thus the merge phase is complete. The reservation grid assigns the vehicle ID with the highest priority (if any) to each tile. Conflicts are resolved for each tile individually according to the assigned priorities.

After collecting and merging the individual reservation requests, the vehicles have to commit to the very same reservation-grid: The final result has to be consistent. Though the participation flags indicate that every node participated in the merging phase at some earlier point, it is not guaranteed that they are all aware of the final schedule; an unmerged request could override some parts. For this reason, we incorporate a second, commit phase. Overall, these transaction semantics are the foundation for safety in STARC: a vehicle can only cross if all others have agreed to the path.

Commit phase. The leader starts the commit phase as soon as it receives the complete schedule. The leader marks the packet as a commit phase transmission and clears all participation flags before transmitting it: The leader commits on the intersection-wide schedule. Further changes are not allowed in the commit phase.

When a node currently in the merge phase receives a commit packet, it switches to the new commit phase and adopts the received reservation-grid, ignoring any local, incomplete state. Its participation flag is set once more, this time as a simple acknowledgment of having received the commit. Nodes retransmit the packet according to the underlying Synchrotron mechanism using the flags as a progress indicator. The commit phase ensures that every node has the chance to participate in the chosen, unique schedule. Due to the retransmissions, almost all are aware of the schedule at the end of the round.

Crossing schedule. Vehicles can check the latest state they received at the end of the round. If it contains a commit, the vehicle verifies the state of its reservation request. The request is only accepted if all tiles along the path have been granted to that vehicle. It follows that every other request either tried to reserve other tiles or had a lower priority. Because the second phase is only started, if all participated, the commit ensures that all requests are merged. They agree on a common intersection-wide schedule.

If a vehicle got accepted, it might mark its reservation as passing for all following rounds. Such a reservation has the highest priority and wins in all conflicts in subsequent rounds. This way, the reservation grid contains both the reservations of the passing and waiting vehicles. Conflicts between passing reservations are impossible as long as the vehicles do not add new tiles to accepted reservations, which is prohibited.

When the vehicle has passed the intersection, it just empties its following requests, and others can reserve those tiles again (the protocol allows freeing unneeded tiles). While issuing more requests seems a little excessive, this mechanism does not require the vehicles to store any states of other vehicles' requests. They could even reconstruct their own state from the position in the intersection. Since there is no need to save the states of the reservations, failures can be handled well, making this part of the protocol fail-safe.

B. Supporting Dynamic Networks

Before a car may participate in a coordination round, it needs to join the network. Its participation is expected as long as the car has joined the network. The car thus needs to leave explicitly. If the car drives off without leaving the network, the commit phase does not start, and the protocol would halt so that progress is impossible.

Leader handover. The leader manages the mapping of Synchrotron indices to vehicle IDs and thus determines which nodes participate with which Synchrotron index. The leader handles joins and leaves and hands-over to another vehicle as soon as it wants to leave the network. As the mapping of the joined nodes is part of that handover, and the payload of a Synchrotron packet is restricted, we limit the network size. As a result, we only allow the first cars of each lane to join the network and assume cars use their sensors to detect cars in front of them.

Join and leave. Nodes may join and leave the STARC network dynamically. A new vehicle starts listening for packets, and with the beginning of the next round, it acts as a forwarder: The node transmits according to the STARC policy and merges partial schedules but is not allowed to issue its own reservation requests until successfully joined. As soon as the vehicle passes the intersection, it leaves the network.

We incorporate the join and leave mechanisms into the coordination round and use monotonically increasing configuration numbers [7]. The nodes compare their locally saved numbers to detect inconsistencies, e.g., missed commits. A node rejoins the network to recover.

C. Platoon Extension

Vehicles that share a common direction may group up and form platoons to utilize the road better and, in our case, the intersection. We can easily extend the STARC primitive to support platoons: While waiting, the vehicles in each lane form a platoon and their platoon head coordinates and reserves the path for the full platoon as if they were a single, long vehicle. The reserved tiles are freed after the last platoon member passes it.

IV. Evaluation

This section analyzes STARC's behavior in demanding wireless environments as well as its efficiency in potential traffic scenarios.

Methodology. The measurements are based on data gained through three 30 minutes simulation runs. Each vehicle performs the following steps: Queueing, waiting for its reservation, moving over the intersection, and leaving the network. The times for queueing, waiting, and leaving determine the delay of the vehicle. We quantify efficiency by the average additional delay and safety by the number of collisions [1]. The fundamental parameter is the number of vehicles per hour that specifies the level of traffic.

Setup. With 16 supported nodes, the network has enough slots to cover all 12 lanes (three lanes per direction) plus four additional slots for crossing or leaving vehicles. Fig. 2a displays the simulated intersection with the corresponding tile-grid. We set the granularity to 6, the resulting tile borders thus match the borders of the lanes. The implementation features four join slots and a single rejoin slot. We set the length of a Synchrotron slot to 6 milliseconds and the number of slots to 200. The interval between two Synchrotron rounds is 2 seconds.

TABLE I. Evaluation parameters

	Parameter	Value
Intersection	Number of lanes	12
	Tile grid	6×6
	Turn rate (left & right)	$15\,\%$ each
	Lane width	$3\,\mathrm{m}$
	Maximum speed	$50\,\mathrm{km\,h^{-1}}$
Vehicle	Acceleration	$2\,\mathrm{m\,s^{-2}}$
	Deceleration	$4\,\mathrm{m\,s^{-2}}$
	Vehicle diameter	$2\,\mathrm{m}$
STARC	Round Interval	$2\,\mathrm{s}$
	Slot length	$6\,\mathrm{ms}$
	Slot number	200
	Max. network size	16
	Join / leave / rejoin slots	4 / 16 / 1

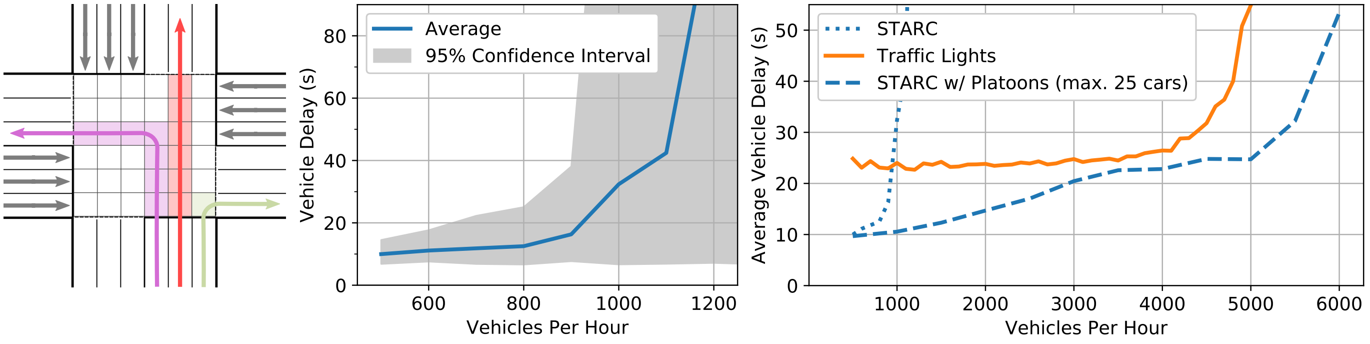
The cars themselves are homogenous and share the same physic specifications, as displayed in Table I. The turning rate is limited to 90 degrees per second. Sharp curves thus require deceleration. As assumed in the design chapter, the vehicles follow a preplanned path to cross the intersection. This path is simplified to match the tiles used in the intersection. The car body is represented by a circle to simplify the simulation and collision checking. The circle has a diameter of 2 meters while the lanes are 3 meters wide. We assume a 15% rate for both right and left turns, 70% of the cars are thus trying to move straight across the intersection. Each starting lane has a corresponding end lane. Lane swaps are prohibited.

Implementation. We use the Synchrotron implementation for TelosB sensor nodes [7]. The TelosB features an MSP430 chipset by Texas Instruments with 10 kB RAM and a 4 MHz CPU [16]. With their 250 kbps IEEE 802.15.4 radio, the nodes act as transmitters for the STARC protocol. They run Contiki OS, an open-source operating system for the Internet of Things [17]. We use the simulator Cooja to emulate the nodes' code execution and simulate their radio messages using a Multipath Ray-tracing radio medium for an almost realistic radio simulation.

A. Radio Failures

Wireless communication is prone to error and unreliable by nature: interference from other technologies, multi-path reflection, and antenna orientation can significantly affect the quality of communication. We evaluate STARC's resiliency against failure and its safety property when some cars are unable to communicate with the rest of the network.

Scenario. We inject failures blocking communication for certain nodes in the network. During a round, at each slot, each node has a probability of failing. Upon failure, the node is unable to communicate: it does neither receive nor transmit any message until the end of the round. All failed nodes



(a) Intersection Layout **(b)** STARC protocol delay without platoons **(c)** STARC and Platoon Extension Compared to Traffic Lights

Fig. 2. *(a)* The simulated intersection has three ways per direction. The arrows represent the movement restrictions with the corresponding tiles in the 6×6 grid. *(b)* The STARC protocol simulations without platoons indicate average delays of less than 20 seconds below 1000 vehicles per hour and congestion afterward. *(c)* While STARC without platoons shows less delay than traffic lights only below 1000 vehicles per hour, the version with platoons dominates even in higher traffic levels.

TABLE II. Commit success and collisions in the presence of radio failures. While failures affect the commit success rates, no collisions occurred in the simulations.

Failure Rate	Commit Success Rate	Car collisions
0 %	99.8%	0
0.001%	99.4%	0
0.01 %	96.1%	0
0.1 %	63.7%	0

recover their communication capabilities once the next round starts. We exclude the leader from failure injection: the leader is always able to communicate. We fix the traffic to 1000 incoming cars per hour.

Results. Table II presents the agreement success rate and the number of car collisions for specific failure rates. Under lower failure rates of up to 0.01%, most rounds are successful, and cars agree on a crossing schedule.

With a failure rate of 0.1%, each car has a probability of 0.1% to stop communicating every 6 ms, and the success rate drops to 63.7%. This means that 6 out of 10 rounds led to an agreement on which cars should cross the intersection. In 4 out of 10 rounds, the network did not commit on a new schedule, but no car disagreed and chose its personal, unsafe schedule: all cars agreed not to cross.

For all failure rates, no collision between cars happened. STARC is safe since it prefers to block crossings to prevent conflicting paths and possible collisions under failures.

B. Intersection Crossing Efficiency

Scenario. We evaluate STARC's efficiency with regards to varying traffic load. Here, we define efficiency as the delay experienced by a user from the time the vehicle reached the intersection up to the point the vehicle leaves the intersection, and the STARC network. We vary the traffic load from very few cars up to 1200 vehicles per hour, which corresponds to a medium city intersection in the morning hours with one car every 3 seconds.

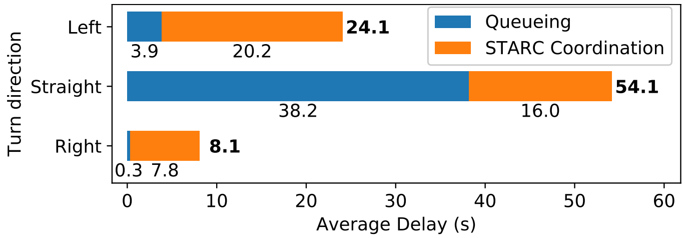


Fig. 3. A delay composition for each turn direction in STARC simulation runs with 1100 vehicles per hour and no platoons. Joining and leaving the network, as well as waiting for the reservation, are part of the coordination. The waiting times correspond to different path lengths. Driving times are excluded. 70% of the cars continue straight.

Results. Fig. 2b depicts the average delay experienced at the intersection running STARC. While the average delay is below 20 seconds at first, the delay spurts upwards at more than 1100 vehicles per hour. However, the lower bound on the confidence interval indicates that some vehicles still experience less delay.

Fig. 3 breaks down vehicle delays for 1100 cars per hour for different directions. With 70% of the vehicles driving straight over the intersection, the straight-driving vehicles spent most of the time (38.2 s) waiting in the queue. This time is lower for vehicles that turn left (3.9 s) or right (0.3 s). The waiting times for acceptance corresponds to the number of tiles needed for the reservation: right-turning vehicles have to wait for the least (6.2 s), while left-turning vehicles have to wait for the longest (18.9 s). The straight-driving vehicles experience intermediate waiting times (14.6 s). The time to leave the network is nearly equal across all directions (1.3 s to 1.5 s).

C. Traffic Lights Comparison

Scenario. The scenario is modified to support traffic lights, but its basics stay the same. When passing over a traffic light intersection, the cars only have two states: queueing and moving. Cars have to queue and wait for a green light before they may cross the intersection. The traffic lights use a simple all-lane-model: All lanes of one direction are allowed to drive simultaneously [1]. Each direction is scheduled once a minute, resulting in 15 seconds per direction. Out of those 15 seconds, a green light is shown for 9 seconds. A yellow and red light shows for 3 seconds each, assuring a safe passing of potential left-turning vehicles. In the scenario with traffic lights, only the time for queueing specifies the delay. We also evaluate the platoon extension of STARC, as described in Section III-C. We limit the platoons to a maximum size of 25 vehicles and only allow joining the platoon before it starts crossing. The platoon head coordinates and reserves the path for the full platoon.

Results. Fig. 2c presents the average delay of traffic lights and the STARC protocol without and with platooning enabled. The platoon size limit of 25 roughly equals the maximum amount of cars that could cross straight during a single green phase. The recorded delay for our protocol without platoons increases slowly, and at more than around 900 vehicles per hour, skyrockets. At around 1000 vehicles per hour, the delay without platoons already surpasses the delay of traffic lights. This limit is not surprising, considering that the protocol without platoons allows only a single vehicle per lane to move. The version with platoons handles those values well and does not only compete with the traffic lights but also further decreases the delay in low traffic settings with less than 1000 vehicle per hour compared to STARC without platoons. All mechanisms can not handle arbitrary high traffic volumes. Just as a longer green phase for the traffic lights, we expect that a higher limit for the platoon size could reduce the experienced delay in higher traffic volumes (without increasing the delay for lower traffic volumes).

V. Conclusion

Crossroads are the bottlenecks of road networks. Intersections often require infrastructure, such as traffic lights, to enable coordinated crossing and ensure the safety of all road users. Connected traffic is a building block of Intelligent Transportation Systems: if vehicles and other road users can communicate, they can coordinate how to efficiently and safely cross intersections. While centralized coordination mechanisms could allow optimal crossings, they require the presence of a central infrastructure, e.g., LTE or 5G coverage.

We introduce STARC, a decentralized reservation-based protocol for safe intersection crossing. STARC uses cheap, low-power wireless radios and builds on top of IEEE 802.15.4 to enable energy-efficient vehicle-to-vehicle communication in the absence of cellular coverage. With STARC, road users can instantly create or join a low-power local network to coordinate at intersections. We show through simulations that STARC is safe, reduces average waiting times by up to 50% compared to traffic lights for volumes lower than 1000 vehicles per hour, and can efficiently serve traffic loads over 1000 vehicles per hour with the support of platoons.

References

[1] Kurt Dresner and Peter Stone. “A Multiagent Approach to Autonomous Intersection Management”. In: *Journal of Artificial Intelligence Research* 31 (Mar. 2008), pp. 591–656.

[2] Masoud Bashiri and Cody H. Fleming. *A platoon-based intersection management system for autonomous vehicles*. 2017, pp. 667–672.

[3] Laurens Hobert, Andreas Festag, Ignacio Llatser, Luciano Altomare, Filippo Visintainer, and Andras Kovacs. *Enhancements of V2X communication in support of cooperative autonomous driving*. Tech. rep. 3. 2016, pp. 27–33.

[4] Mark Van Middlesworth, Kurt Dresner, and Peter Stone. *Replacing the stop sign: Unmanaged intersection control for autonomous vehicles*. Tech. rep. 2008, pp. 1381–1384.

[5] Michel Ferreira, Ricardo Fernandes, Hugo Conceição, Wantanee Viriyasitavat, and Ozan K. Tonguz. “Self-organized traffic control”. In: *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM* (2010), pp. 85–89.

[6] Falko Dressler, Christoph Sommer, David Eckhoff, and Ozan K. Tonguz. *Toward realistic simulation of intervehicle communication*. Tech. rep. 3. 2011, pp. 43–51.

[7] Beshr Al Nahas, Simon Duquennoy, and Olaf Landsiedel. “Network-wide consensus utilizing the capture effect in low-power wireless networks”. In: *SenSys 2017 - Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems*. Vol. 2017-Janua. 2017, 1DUMMY.

[8] Michele Segata, Bastian Bloessl, Stefan Joerer, Christoph Sommer, Mario Gerla, Renato Lo Cigno, and Falko Dressler. “Toward communication strategies for platooning: Simulative and experimental evaluation”. In: *IEEE Transactions on Vehicular Technology* 64.12 (2015), pp. 5411–5423.

[9] Hanwen Cao, Sandip Gangakhedkar, Ali Ramadan Ali, Mohamed Gharba, and Josef Eichinger. *A 5G V2X testbed for cooperative automated driving*. 2017.

[10] Christoph Sommer, Florian Hagenauer, and Falko Dressler. *A networking perspective on self-organizing intersection management*. 2014, pp. 230–234.

[11] Michel Ferreira and Pedro M. D’Orey. “On the impact of virtual traffic lights on carbon emissions mitigation”. In: *IEEE Transactions on Intelligent Transportation Systems* 13.1 (2012), pp. 284–295.

[12] Florian Hagenauer, Patrick Baldemaier, Falko Dressler, and Christoph Sommer. *Advanced Leader Election for Virtual Traffic Lights*. Tech. rep. 1. 2014, pp. 11–16.

[13] Abdallah A. Hassan and Hesham A. Rakha. “A Fully-Distributed Heuristic Algorithm for Control of Autonomous Vehicle Movements at Isolated Intersections”. In: *International Journal of Transportation Science and Technology* 3.4 (2014), pp. 297–309.

[14] R. Naumann, R. Rasche, J. Tacken, and C. Tahedl. “Validation and simulation of a decentralized intersection collision avoidance algorithm”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. 1997, pp. 818–823.

[15] Olaf Landsiedel, Federico Ferrari, and Marco Zimmerling. “Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale”. In: *SenSys 2013 - Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems* (2013).

[16] *MSP430 Ultra-Low-Power MCUs*. URL: http://www.ti.com/microcontrollers/msp430-ultra-low-power-mcus/overview.html.

[17] *Contiki: The Open Source Operating System for the Internet of Things*. URL: http://contiki-os.org/.