Real-time Decentralized Monocular SLAM

Guillaume Bresson, Romuald Aufrère, Roland Chapuis

Real-time Decentralized Monocular SLAM

Guillaume Bresson*, Romuald Aufrère*† and Roland Chapuis*
*Institut Pascal - UMR 6602 CNRS – †LIMOS - UMR 6158 CNRS
Clermont Université, Université Blaise Pascal - Aubière, France
firstname.name@univ-bpclermont.fr

***Abstract*—This paper presents a real-time Decentralized Monocular SLAM process. It is the first time, to our knowledge, that a decentralized SLAM with vehicles using only proprioceptive sensors and a single camera is presented. A new architecture has been built to cope with the problems involved by a decentralized scheme. A special care has been given to the data incest problem. It is solved thanks to a substate system. Network aspects and computational time are also considered. By using an Extended Kalman Filter and sending only essential information, we are able to make our decentralized algorithm suitable for an important number of vehicles. We also introduce a way to retrieve the distance between vehicles and so to put the different maps built in a common frame. This approach was tested using a realistic simulator with different trajectories.**

I. Introduction

The Simultaneous Localization And Mapping (SLAM) problem has received a significant attention during the past twenty years [8]. Solutions relying on different methods and sensors have been proposed [1]. However, little focus has been given to the multi-robot case while many applications require a team of vehicles. Fast exploration [10], ground/aerial cooperation [17], fleet localization without direct observation [13]... are examples of tasks only possible with several robots operating together. Nevertheless, making vehicles cooperate raises a lot of issues that need to be addressed. Data exchange strategy, network aspects, information fusion are key elements that must be considered in relation to robustness, efficiency and cost. The choices involved here, mostly depend on the application aimed. We intend to provide a localization algorithm able to give the best pose estimate of each vehicle of a team without necessarily seeing each other.

The first, and certainly most important, choice is the SLAM architecture. A major part of multi-robot SLAM algorithms are using a centralizing entity [11][17]. It has the advantage to make the process quite straightforward. Each vehicle sends sensors information or higher-level landmarks to a single entity (meaning a vehicle or a dedicated computer) which agglomerates all the data in order to perform a standard SLAM process. Though simple, this solution is not robust as a failure of the centralizing node will prevent the system from working. Moreover, it also limits the distance that vehicles can cover as they must stay within reach of the centralizing entity. To be truly robust, the system must be fully decentralized. This means that there is no leader in the team on which everyone's localization is based [9]. The decentralized scheme has to deal with data incest. Indeed, as each robot is sending and receiving information to and from other vehicles, a special attention must be given to avoid double counting information.

The decentralized scheme involves also an important network aspect. This point is often neglected [19]. However, it is a mandatory step for realistic experiments. A distributed algorithm must handle period latencies, desynchronized data and even communication losses. Still concerning network, bandwidth requirement can be an important issue. Indeed, to be truly scalable, a decentralized system should be independent of the number of vehicles [14]. Nevertheless, it is restrictive as it forces the use of point-to-point communications consequently leading to a slower information spread (each node must transmit the information to its neighbor and so on). To avoid this drawback, we decided that our system must be able to scale nicely with the number of vehicles. It means that the quantity of information sent through the network must be as low as possible.

Many SLAM algorithms are based on hybrid maps [6]. Classical metric submaps (or occupancy grids) are built while a higher-level map (topological) keeps track of the starting points of all these submaps so as the links between them. This approach has been widely used and has also been extended to the multi-vehicle case during the last years [19]. Indeed, it can be done quite easily while solving the data incest problem. It also allows to keep a low exchange rate through the network as only the topological map is sent. In [17], for instance, the authors exchange the topological map and some information to perform associations. Once a landmark, existing in a map, is found in another one, a link is added between those two submaps and their initial starting point is refined in the topological map. However, the metric submaps are not improved as it would be a time consuming process (it is usually done afterwards). Consequently, navigating through a submap will not be accurate even though it could be. Switching between submaps becomes more difficult as the accurate pose of the vehicle is not available. It could potentially lead to wrong detections and to a failure of the SLAM process. More generally, using only submaps can be a problem as the information about them is only available once they are closed. For instance, if two vehicles start navigating in a row, it could be useful to fuse landmarks from the other vehicle as soon as they are available to help the localization process.

Most decentralized SLAM solutions have only been tested under restrictive simulation constraints (e.g. no network latencies...). Moreover, they mainly rely on range and bearing sensors that are expensive. These sensors also rarely furnish enough distinguishable features which is a major issue in a decentralized process. On the other hand, cameras have been

used a lot to recognize previously seen features [18]. To our knowledge, only two approaches have used these sensors for multi-robot SLAM [11][17]. In both cases, it was stereo vision and only in a centralized fashion.

In this paper, we propose a solution to the decentralized SLAM problem. Our approach relies on vehicles equipped with only one camera and proprioceptive sensors (giving odometric and steering information). The decentralized scheme presented here fuse all incoming data in real-time in order to have an accurate global map available at any time. Data incest is avoided by our exchange strategy and the bandwidth and processing time required are low compared to the current technological possibilities. Network latencies and desynchronizations are also addressed.

This paper will be organized as follows: Section II will present the Monocular SLAM in a single robot context. Section III-A will explain the decentralized data fusion process in an ideal case. Then, Section III-B will take into account the network issues previously exposed. Section IV will present the experiments realized thanks to a realistic simulator (3D environments with a realistic physics). The results of the experiments conducted will eventually be detailed.

II. Monocular SLAM

Before starting to decentralize the SLAM process, it is necessary to build a robust single vehicle algorithm able to match the constraints induced by the multi-robot case. Using an Extended Kalman Filter (EKF), a Bundle Adjustment based method (BA) or a Particle Filter (PF) makes a huge difference on the whole system. Every method has drawbacks and can be time consuming. However, EKFs have the advantage to be light with only a few landmarks needed to obtain an accurate localization estimate. This is a key aspect as it means that the bandwidth required will be smaller than with BAs for example, where hundreds of points are needed. Moreover, a direct access to a landmark covariance matrix is important for a robust data association process. However, with a bearing-only sensor, the linearization errors bound to the important uncertainties are an issue that must be addressed. Consequently, feature initialization must be treated with a special attention.

Indeed, with the need of a global map as accurate as possible, it is extremely important to deal with landmarks initialization and the way they are handled throughout the process. With a single camera, only the bearing of the landmark is available. Several observations with enough parallax are needed to get an accurate position.

Two solutions can be found in the literature: delayed and undelayed initializations. The first one uses only accurate landmarks [7]. They are kept outside of the estimation process until they are sufficiently precise. However, there is an important loss of information. For instance, during a bend, features will pass out of the field-of-view quickly without having the time to be added to the state vector though carrying useful bearing information. Conversely, undelayed approaches integrate landmarks since their first observation. Positions are then refined within the estimation process. However, as their depth is unknown, linearization issues will quickly appear when tracking them or updating their position [2]. Different landmark representations have been used to counter this effect. In [15], multiple hypotheses are created and integrated into the state vector. With new observations, wrong hypotheses will be discarded. An extra-cost is involved by the necessity to treat several hypotheses for a single landmark. The Inverse Depth (ID) parametrization is the most widely utilized representation [5]. 6 parameters are needed instead of 3 to estimate a single 3D point. It can be computationally expensive since the size of the state vector is duplicated. Instead, we chose to focus on a simple Cartesian representation [3] but to address the linearization errors involved.

During the tracking step, the landmark covariance is projected onto the image thanks to the computation of a proper bounding box [3] making the matching process free from linearization errors. For the update step, a corrective factor of the Kalman gain is computed in order to drastically reduce linearization issues [4]. Consequently, our SLAM algorithm is fast and memory thrifty allowing to extend it more easily to the multi-vehicle case. Combined with an EKF, only a few landmarks are needed to achieve good localization results making it ideal for decentralized SLAM.

III. Decentralized Data Fusion

A. Ideal case

When dealing with DDF (Decentralized Data Fusion), one of the major problems is to avoid data incest. It occurs when a vehicle double counts an information. For example, a vehicle sends the localization of a landmark to another which resends it to the emitter. Without a special care during the fusion process, the information will be fused again reducing the uncertainty of the estimate thus losing the integrity.

Data incest can be avoided thanks to a dedicated network architecture. In [14], each vehicle only sends information to its neighbor. A trace of every communicated data is kept thus preventing from double counting information. However, it also means that two vehicles located at opposite part of the network will have to pass by every other node to eventually reach each other. Dedicated architectures put aside, it is possible to deal with data incest by never exchanging fused information. Each vehicle performs its own fusion. Given that every vehicle has received the same information, fused maps are equivalent. In [12], the authors consider that each vehicle's state is a substate. The substates are communicated to all the vehicles of a team. Fusing all the substates gives the global state which gathers all the available information. The main advantage of this method is that, when a new substate is received, it replaces the old one, avoiding data incest while still having the best estimate. Applied to a SLAM context, it can be time consuming because each landmark already received will be replaced by the new one, forcing us to break the previously built global map to re-build it with the new information.

In this paper, we use a similar approach but adapted to a SLAM process. Each vehicle performs its own SLAM and builds its own map. A decentralized process (ran by every

vehicle) gathers the different substates (the local one and those received through the network) and fuses them. As in [12], a vehicle pose is replaced in the corresponding substate each time a new one is received. To avoid having to break and re-build the whole map, we decided to only exchange accurate landmarks. It means that when the uncertainty of a landmark falls below a threshold, it is copied once and for all in the decentralized part of the SLAM (and sent to the other vehicles). The modifications brought within the single vehicle monocular SLAM process will not be used in the decentralized system. It is not a problem since landmarks are accurate. The advantage is that only new landmarks will be added to the corresponding substate. The fused state (global map) will always be kept and enhanced with incoming landmarks (local and coming from other vehicles). Associations found in the global map are not re-injected in the local map thus avoiding data incest. Thanks to this method, the local SLAM does not have to keep the landmarks when they are not visible anymore. This task is handled by the decentralized process. As a consequence, the local SLAM will be fast. Figure 1 shows the interactions between the local and decentralized parts of the SLAM process. Figure 2 illustrates how the substates are handled within the decentralized algorithm.

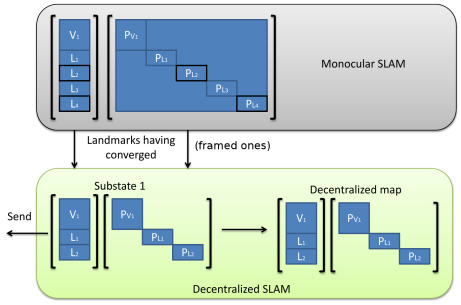


Fig. 1. Interactions between local and decentralized processes in a vehicle

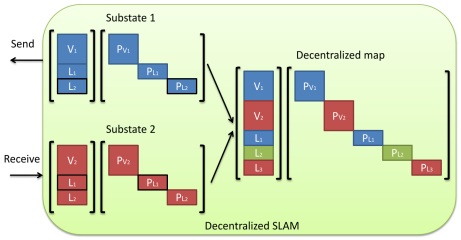


Fig. 2. Substates handling and global map (common landmark in green)

The main objective of fusing landmarks coming from different vehicles is to retrieve the link between those vehicles. Indeed, without a sensor giving a global estimate, putting different maps in a common frame is a difficult task. Authors generally use a GPS to have an idea of the distance between the different robots [17]. Other approaches rely on a rendezvous to obtain this estimate [20]. In this paper, we do so by fusing landmarks from different vehicles. However, it can be difficult without a prior knowledge about where the vehicle is (or approximately is) with relation to a global frame. We propose here an elegant way to integrate this prior information and its corresponding uncertainty. Each vehicle expresses landmarks in its own frame. However, in a decentralized architecture, the goal is to have a common frame. We chose to model the link between the local frame and the common one as a bias. At the beginning, it can be modeled with an important uncertainty. Finding common landmarks between two vehicles will help reducing this uncertainty and estimate the distance (and angles) between the two frames. Figure 3 gives an example with a bias allowing to express one vehicle in the other one's frame. The bias allows a proper transformation to a common frame.

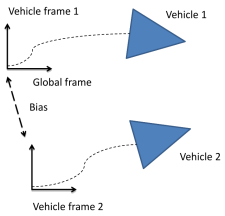


Fig. 3. Bias allowing to express one vehicle in the other one's frame

Using the bias has another advantage. The vehicle pose is accurate within its own local frame. However, in the global frame, it is not the case while no landmarks have been associated. It means that almost all the uncertainty of the state vector is bound to this lack of knowledge about the global frame. We can use the bias to find the links between the vehicles and landmarks as they mostly depend of it. Consequently, we can limit the network burden by only sending landmarks and vehicles variances and not the cross-covariances. It greatly reduces the size of the data sent through the network without affecting the decentralized process.

Let $\mathbf{b}_i$ be the bias applied to the vehicle i allowing it to be expressed in the global frame G. It is defined as follows:

$$\mathbf{b}_i = \begin{pmatrix} b_{x_i} & b_{y_i} & b_{z_i} & b_{\alpha_{x_i}} & b_{\alpha_{y_i}} & b_{\alpha_{z_i}} \end{pmatrix}^T$$

where b_{x_i}, b_{y_i} and b_{z_i} represent a position bias and $b_{\alpha_{x_i}}$, $b_{\alpha_{y_i}}$ and $b_{\alpha_{z_i}}$ an angular bias. Let $\mathbf{v}_i$ be the vehicle in its own reference frame:

$$\mathbf{v}_i = \begin{pmatrix} v_{x_i} & v_{y_i} & v_{z_i} & v_{\alpha_{x_i}} & v_{\alpha_{y_i}} & v_{\alpha_{z_i}} \end{pmatrix}^T$$

Regarding to G, the bias is additive and we can write:

$$\mathbf{v}_{G_i} = \mathbf{R}\mathbf{v}_i + \mathbf{b}_i \tag{1}$$

where $\mathbf{R}$ is:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{3D}(b_{\alpha_{z_i}}, b_{\alpha_{y_i}}, b_{\alpha_{x_i}}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix}$$

with $\mathbf{R}_{3D}$ being a classical product of 2D rotation matrices. Similarly, we can define the transformation from the local frame to the global one for landmarks. Let $\mathbf{l}_{ij}$ be the j^{th} landmark of the i^{th} vehicle:

$$\mathbf{l}_{ij} = \begin{pmatrix} l_{x_{ij}} & l_{y_{ij}} & l_{z_{ij}} \end{pmatrix}^T$$

The transformation to G is:

$$\mathbf{l}_{G_{ij}} = \mathbf{R}_{3D}\mathbf{l}_{ij} + \begin{pmatrix} b_{x_i} \\ b_{y_i} \\ b_{z_i} \end{pmatrix} \tag{2}$$

Equations 1 and 2 can be expressed together through a function f_i taking the incoming vehicle and landmarks with the bias:

$$\begin{pmatrix} \mathbf{v}_{G_i} \\ \mathbf{l}_{G_{ij}} \\ \dots \end{pmatrix} = f_i(\mathbf{v}_i, \mathbf{l}_{ij}, \dots, \mathbf{b}_i) \tag{3}$$

It is then possible to obtain the associated covariance matrix $\mathbf{P}_{G_i}$ by computing the associated Jacobian:

$$\mathbf{P}_{G_i} = \mathbf{J}_{f_i} \mathbf{P}_i \mathbf{J}_{f_i}^T \tag{4}$$

Through this process, the bias will naturally retrieve the missing links between vehicle and landmarks and between landmarks themselves. To do so, the Jacobian can be split as the parameters of f_i are independent at the moment:

$$\mathbf{P}_{G_i} = \mathbf{J}_{f_{v_i}} \mathbf{P}_{v_i} \mathbf{J}_{f_{v_i}}^T + \mathbf{J}_{f_{l_{ij}}} \mathbf{P}_{l_{ij}} \mathbf{J}_{f_{l_{ij}}}^T + \dots + \mathbf{J}_{f_{b_i}} \mathbf{P}_{b_i} \mathbf{J}_{f_{b_i}}^T \tag{5}$$

Computing those Jacobians is straightforward and will not be detailed here for space purposes. If previous landmarks coming from the i^{th} vehicle have already been added to the global map, it is also necessary to reestablish their links with the incoming landmarks. It is done by recomputing the Jacobians for those specific points. After the fusion process, with a 2-vehicle example, the decentralized state can be seen as in Figure 4.

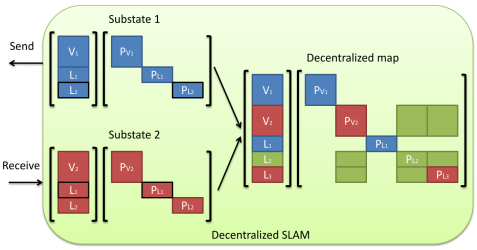


Fig. 4. After a fusion between landmarks (common information in green)

Landmarks that have not been associated with those already in the global state are directly added to it (by taking into account the bias). The others are fused with a simple Kalman filter. An association table keeps track of all the associations found between the different substates and to which fused landmark they refer in the global map.

A standard matching approach is used to associate points from different vehicles. It consists of a Mahalanobis distance criterion followed by a feature correlation. Thanks to the bias, the Mahalanobis distance is valid but not very restrictive, especially at the beginning of a trajectory when the bias is uncertain. To avoid as many wrong associations as possible, the descriptors (11×11 pixels extracted around Shi-Tomasi features) used by the monocular SLAM are also exchanged. By comparing two descriptors (via Normalized Cross Correlation), it is possible to confirm (or not) the association found by the Mahalanobis process.

B. Realistic case

In real life applications, some aspects must be considered with a special attention. Each sensor delivers information at a different speed. In a decentralized system, it could become even more difficult. However, it is easy to understand that an approach based on raw sensor information exchange is not viable as it would mean a constant communication leading to a saturated network. In the proposed approach, this problem is relegated to the local SLAM. Proprioceptive data is used to predict the state and exteroceptive to update it via an observation model. To deal with desynchronized data, when an image is available, the state is extrapolated to the current time thanks to an evolution model independent of the odometry. The two state estimates are then fused later, with an odometric data. Figure 5 shows an example of how desynchronized information is handled. In the decentralized SLAM, vehicles must be synchronized with the same time line. This can easily be done using a NTP server. A new vehicle pose is always received with an associated time. Thanks to a constant velocity model, it is possible to have an estimate of every vehicle at the same time and consequently to fuse landmarks properly.

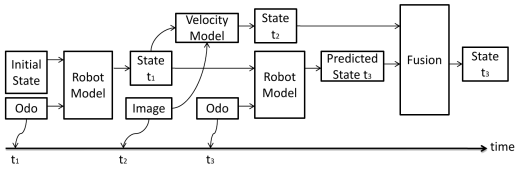


Fig. 5. Handling of desynchronized information

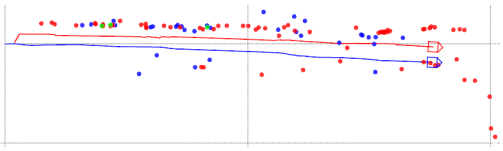
The different observations can be affected by latencies. It can also be an issue as it means that the current state is not the best estimate. In the local process, a dating system (with an absolute reference) for the observations is utilized. Thanks to it, observations are reorganized chronologically. Delayed observations will force to re-compute the state from the late observation till the current time. This problem is treated with more details in [16]. In the decentralized process, latencies are avoided thanks to the exchange strategy. Only a single accurate position per landmark is sent avoiding thus to have to deal with latencies as no fusion is needed. For example, if a landmark is received a second time (relayed by a third vehicle), it will simply not be considered. To do so, each landmark comes with a unique vehicle index and a unique landmark index. For vehicles, if the state sent is older than the current one, it is not considered. It is not a problem because each vehicle pose sent is the best estimate to date. Consequently, no information is lost. The unique index has another advantage. The index is incremented each time a new landmark is added to the local substate. When receiving new landmarks, the decentralized process checks if there are missing ones in the corresponding substate. If it is the case, a request frame can be sent to the concerned vehicle. This simple process can cope with data losses.

IV. Experiments

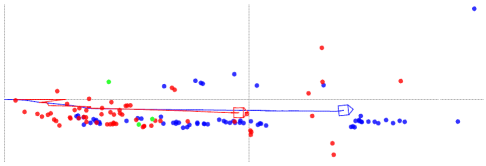
The experiments presented in this paper have all been conducted thanks to a simulator for convenience purposes. However, the simulator used presents a realistic physics and has several environments corresponding to real locations such

as our hometown town center (which is the one used in these experiments). The simulator is only used in order to generate sensor information. The processing corresponding to each vehicle is executed on different computers with a real network communication. Several pictures illustrating the environment used and a typical camera output can be shown in Figure 6. Two trajectories, with two vehicles each, were accomplished. Both vehicles were equipped with a single camera running at 10 Hz and with a proprioceptive sensor giving odometric and steering information. The vehicles were going at approximately 2 meters per second. The Monocular SLAM used a maximum of 20 landmarks per image and only those having converged are copied in the decentralized algorithm.

The first trajectory consisted of two vehicles, side by side, moving forward. The idea was to see if, without seeing each other, the vehicles were capable of retrieving the link between them solely by finding common landmarks. It allowed to verify how fast the bias can converge towards its true value. We used a bias equals to zero as an initial value but with a sufficient uncertainty to include the global frame. The same settings were applied to the second trajectory. This one started with two vehicles aligned in a column but separated by several meters. Here, we wanted to check if it was possible for two vehicles evolving in a convoy to find common landmarks. These two trajectories can be seen in Figure 7.



(a) First trajectory



(b) Second trajectory

Fig. 7. In blue: landmarks and vehicle trajectory of the first vehicle. In red: same thing for the second one. In green: common landmarks.

Figure 7 shows only the landmarks which have converged in the monocular SLAM. It corresponds to those which are exchanged in the decentralized process. In both experiments, the objective was to express the red vehicle pose (and landmarks) in the blue one's frame. Associated landmarks helped refining the bias. It can be noticed in Fig. 7 that the sudden changes in the trajectory correspond to new matchings found and therefore to a bias refinement. Only a few matchings are found in both experiments (3 in each). This can easily be explained by two main reasons. The first one is that data association is difficult as the vehicle pose is very uncertain at the beginning. To avoid false associations, we used discriminant thresholds in both experiments. While it avoids false data association, it also prevents good ones from being performed. A more robust process would help in this case. The second reason is bound to the fact that the bias estimated here is static. It means that once a few associations are found, the uncertainty will be close to zero. Finding matchings becomes almost impossible. Moreover, it is well-known that the SLAM process drifts as time passes causing a loss of integrity. In a decentralized case, each vehicle drifts differently from the others meaning that detecting new associations is even more complicated. By modeling this phenomenon, it would be possible to prevent it.

Tables IV and II show the evolution of the bias for the different associations found compared to the true distance between the two vehicles at the beginning of the experiments (root mean-square error). The difference for the position (x, y and z) is expressed in meters and for the angles (α_x, α_y and α_z) in radians. We can notice that with new landmarks associated, the bias tend to be closer to its true value. Sometimes it slightly diverges because of landmarks less accurate than others.

	x	y	z	α_x	α_y	α_z
Matching 1	0.1679	0.0571	0.0025	0.0002	0.0006	0.0026
Matching 2	0.0519	0.0970	0.0007	0.0017	0.0002	0.0052
Matching 3	0.0435	0.1215	0.0255	0.0077	0.0011	0.0019

TABLE I
BIAS ERROR FOR THE DIFFERENT MATCHINGS (FIRST TRAJECTORY)

	x	y	z	α_x	α_y	α_z
Matching 1	3.1282	1.0247	0.2300	0.0369	0.0110	0.0815
Matching 2	1.9790	0.3605	0.2357	0.0168	0.0203	0.0113
Matching 3	0.3123	0.2599	0.2081	0.0152	0.0207	0.0074

TABLE II
BIAS ERROR FOR THE DIFFERENT MATCHINGS (SECOND TRAJECTORY)

Concerning bandwidth requirements and computational time, the choice of an EKF gives excellent results. Figure 8 indicates the computational time of our algorithm. As the decentralized process is fully multi-threaded, the different parts (sending, receiving and fusing information) are executed independently. The computational time given in Figure 8 concerns the second trajectory (results are similar for the first one). Times are given for each call made which means that the algorithm is not active most of the time. The worst case noticed here barely goes above 100 μs.

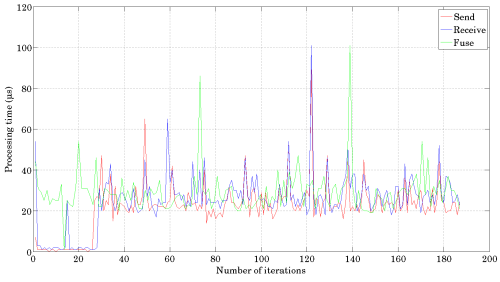


Fig. 8. Computational time required by the decentralized process

The bandwidth requirements are given in Figure 9. For both vehicles, less than 2 KB are sent per second. These results (bandwidth and computational time) show that our approach could easily be used with a very important number of vehicles.

(a) Environment used (b) First trajectory (c) Second trajectory (d) Camera output

Fig. 6. Different illustrations of the simulator.

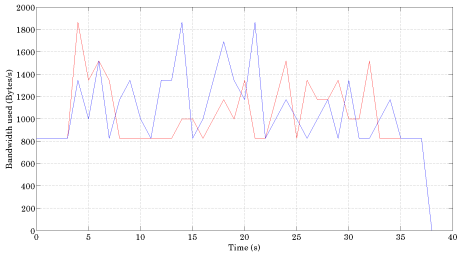


Fig. 9. Data sent through the network per second by the first vehicle (blue) and the second one (red).

V. Conclusion

A Decentralized Monocular SLAM has been presented. It is the first time, to our knowledge, that a decentralized SLAM using only vehicles equipped with a single camera and proprioceptive sensors is presented. Data incest is avoided thanks to the substate system. Each vehicle fuses the available information but never share the fused state. By exchanging landmarks, it is possible to have a direct estimate of where the other vehicles are located instead of having to wait for a submap to be closed.

In order to put the maps in a global frame, we introduced the bias. Its goal is to represent the lack of knowledge about the inter-vehicle distance. It allows to find associations between landmarks and therefore to progressively refine its estimation.

We have evaluated our approach over two trajectories. With several associations, we were able to retrieve the distance between the vehicles. The two sequences illustrate that our approach can work with vehicles evolving side by side or in a convoy. The time required by the decentralized process is negligible (less than 100 μs). The bandwidth used is also very small remaining under 2 KB per second. Consequently, our approach scales nicely with the number of vehicles.

However, a more robust data association will be investigated to avoid wrong associations. Using a GPS could also be a way to restrict the bias and ease the association process. Another perspective is consider a dynamic bias which would help dealing with the SLAM drift.

References

[1] T. Bailey and H. Durrant-Whyte. Simultaneous Localization and Mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006.

[2] K. E. Bekris, M. Glick, and L. E. Kavraki. Evaluation of Algorithms for Bearing-Only SLAM. In *IEEE International Conference on Robotics and Automation*, pages 1937–1943, 2006.

[3] G. Bresson, T. Féraud, R. Aufrère, P. Checchin, and R. Chapuis. A New Strategy for Feature Initialization in Visual SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Perception and Navigation for Autonomous Vehicles in Human Environment*, pages 115–120, 2011.

[4] G. Bresson, T. Féraud, R. Aufrère, P. Checchin, and R. Chapuis. Parsimonious Real Time Monocular SLAM. In *IEEE International Conference on Intelligent Vehicles*, pages 511–516, 2012.

[5] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse Depth Parametrization for Monocular SLAM. *IEEE Transactions on Robotics*, 24(5):932–945, 2008.

[6] L. A. Clemente, A. J. Davison, I. D. Reid, J. Neira, and J. D. Tardós. Mapping Large Loops with a Single Hand-Held Camera. In *Robotics: Science and Systems*, 2007.

[7] A. J. Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *IEEE International Conference on Computer Vision*, pages 1403–1410, 2003.

[8] H. Durrant-Whyte and T. Bailey. Simultaneous Localization and Mapping: Part I. *IEEE Robotics and Automation Magazine*, 13(2):99–110, 2006.

[9] T. Féraud, P. Checchin, R. Aufrère, and R. Chapuis. Communicating Vehicles in Convoy and Monocular Vision-based Localization. In *7th Symposium on Intelligent Autonomous Vehicles*, volume 7, 2010.

[10] C. M. Gifford, R. Webb, J. Bley, D. Leung, M. Calnon, J. Makarewicz, B. Banz, and A. Agah. Low-Cost Multi-Robot Exploration and Mapping. In *IEEE International Conference on Technologies for Practical Robot Applications*, pages 74–79, 2008.

[11] A. Gil, O. Reinoso, M. Ballesta, and M. Juliá. Multi-robot visual SLAM using a Rao-Blackwellized particle filter. *Robotics and Autonomous Systems*, 58(1):68–80, 2010.

[12] N. Karam, F. Chausse, R. Aufrère, and R. Chapuis. Localization of a Group of Communicating Vehicles by State Exchange. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 519–524, 2006.

[13] H. S. Lee and K. M. Lee. Multi-Robot SLAM Using Ceiling Vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 912–917, 2009.

[14] E. Nettleton, S. Thrun, H. Durrant-Whyte, and S. Sukkarieh. Decentralised SLAM with Low-Bandwidth Communication for Teams of Vehicles. In *Field and Service Robotics*, pages 179–188, 2006.

[15] J. Solà, A. Monin, M. Devy, and T. Lemaire. Undelayed Initialization in Bearing Only SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2499–2504, 2005.

[16] C. Tessier, C. Cariou, C. Debain, F. Chausse, R. Chapuis, and C. Rousset. A Real-Time, Multi-Sensor Architecture for Fusion of Delayed Observations: Application ot Vehicle Localization. In *IEEE Conference on Intelligent Transportation Systems*, pages 1316–1321, 2006.

[17] T. A. Vidal-Calleja, C. Berger, J. Solà, and S. Lacroix. Large Scale Multiple Robot Visual Mapping with Heterogeneous Landmarks in Semi-structured Terrain. *Robotics and Autonomous Systems*, In Press, Corrected Proof:–, 2011.

[18] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós. A comparison of loop closing techniques in monocular SLAM. *Robotics and Autonomous Systems*, 57(12):1188–1197, 2009.

[19] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte. Towards Multi-Vehicle Simultaneous Localisation and Mapping. In *IEEE International Conference on Robotics and Automation*, pages 2743–2748, 2002.

[20] X. S. Zhou and S. I. Roumeliotis. Multi-robot SLAM with Unknown Initial Correspondance: The Robot Rendezvous Case. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1785–1792, 2006.