CoMapping: Multi-robot Sharing and Generation of 3D-Maps applied to rural and urban scenarios

Luis F. Contreras-Samamé, Salvador Domínguez-Quijada, Olivier Kermorgant and Philippe Martinet

***Abstract*— We present an experimental study for the generation of large 3D maps using our CoMapping framework. This framework considers a collaborative approach to efficiently manage, share, and merge maps between vehicles. The main objective of this work is to perform a cooperative mapping for urban and rural environments denied of continuous-GPS service. The study is split in to 2 stages: Pre-Local and Local. In the first stage, each vehicle builds a Pre-Local map of its surroundings in real-time using laser-based measurements, then relocates the map in a global coordinate system using just the low cost GPS data from the first instant of the map construction. In the second stage, vehicles share their pre-local maps, align and merge them in a decentralized way in order to generate more consistent and larger maps, named Local maps. To evaluate performance of all the cooperative system in terms of map alignments, tests are conducted using 3 cars equipped with LiDARs and GPS receiver devices in urban outdoor scenarios of the École Centrale Nantes campus and rural environments.**

I. Introduction

Urban and rural outdoor scenarios can be a real challenge to perform mapping, since environment complexity, such as terrain roughness or lack of structure or dimensions can affect negative the task performance. Other aspects as access to unexplored area, dimension of the region or communication constraints can be considered in the process. In the case of constructing maps for large areas, the idea of using a set of vehicles for building accurate maps in a reasonable amount of time is feasible [1], because a cooperative mapping extends the capability of a single robot by sharing and merging data between group members. When all the data is analysed and merged in a single computation unit, the process is called centralized multi-robot mapping [2]. On the opposite, the goal here is to have each mobile unit build its own map and merge them upon rendezvous, which is a decentralized process [3], [1], [4].

This approach is considered in this article in order to obtain the 3D map of an urban environment. Two vehicles, the ZOE and FLUENCE, work independently during the exploration and define a meeting point to allow direct exchange of data (such as pose, size, limits and maps) during the map reconstruction. The mapping of a rural scenario was also considered, in which was used Lidar-measurement data performed by GOLFCAR robot. The mapping was based on multi-robot approach because the GOLFCAR path was split in two, simulating two robots for the task (see Fig. 1)

Luis Contreras, Salvador Domínguez and Olivier Kermorgant are from the Laboratoire des Sciences du Numérique de Nantes (LS2N), École Centrale de Nantes (ECN), 44300 Nantes, France. Emails: {name.surname}@ls2n.fr.

Philippe Martinet is with INRIA Sophia Antipolis, 06902 Sophia Antipolis, France Email: Philippe.Martinet@inria.fr

This work was supported by the Erasmus Sustain-T and ALFS project. Parts of the devices used here were funded by the project ROBOTEX, reference ANR-10-EQPX-44-01.

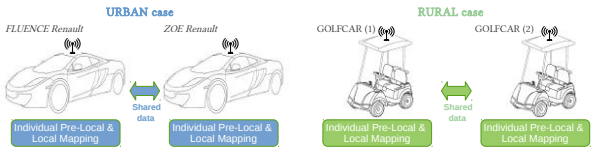


Fig. 1. Scheme of the CoMapping System for different cases

Following this direction, we present the development and validation of a Cooperative Mapping framework (CoMapping) applied to outdoor scenarios based on 2 stages, where:

- In the first stage named "Pre-Local Mapping", each platform constructs its own map by processing range measurements from a 3D LiDAR for a six degrees of freedom (6-DOF) movement and using GPS data (GPS/GGA) only in the beginning for representing the map in robot common frame.
- In the second stage named "Local Mapping", the robots define which part of their pre-local maps are shared with the other robots based on a Sharing algorithm from the CoMapping framework. Finally the registration process is executed, including an intersecting technique of maps to accelerate the process.

We also propose some indicators to analyse the alignment process in both stages. Our proposal has been tested and validated in real situations. The results include maps developed with data acquired on the surroundings of the ECN (École Centrale Nantes) campus for the urban environment and a farm for the rural experiment, corresponding to the ALFS project.

II. Related works

An important topic of the mapping problem is the type of representation used for the environment. Generally 3 types of representations are encountered: feature, grid and topological maps [5]. Our system uses two types of map representation: 3D point cloud (feature) for the pre-processing in Pre-Local Mapping and Octree format (grid) for the exchange of maps in the Local Mapping stage later.

In a situation of collaborative mapping, the registration method from a single robot is really important. Several registration applications use Laser Range-finder sensor to build 3D maps [6] [7]. In that case, a high laser scan rate compared to its tracking can be harmful for this task, since it may lead to distortion in the map construction, in which methods based on Iterative Closest Point (ICP) [8] can be used to match laser returns for different scans. Implementations, with 2-axis and 3-axis lidar and matches geometric structures of a set of local point generated to finally get a point cloud, were presented in [9]. In all those cases, the proposed methods used batch processing to build

the maps with accuracy, hence are not applicable to real-time map construction. Regarding our Pre-Local Mapping stage, we reconstruct the map of the environment as a point cloud in real-time using a 3-axis lidar by extraction and matching of geometric features in Cartesian space based on a modified version of the registration method from [10]. Then our system uses GPS position data to re-localize the cloud in a global frame.

Several methods have been proposed to merge maps. In [2] and [11] techniques were proposed for 3D merging of occupancy grid maps based on octrees [12]. Their merging process refined the transformation estimate between maps by ICP registration [8]. Specifically, in [11] an ICP version was performed including an efficient technique to exchange maps between robots in order to optimize the bandwidth resources of a multi-robot network for decentralized cases. Those last cases are experimentally studied in this paper.

A. Merging Indicators

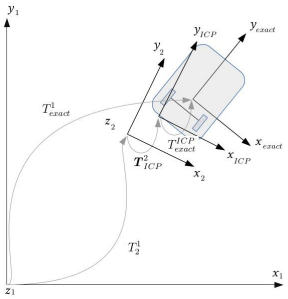


Fig. 2. Transformation Matrices Relations

To evaluate the alignment post-ICP, we have first to determine the transformation matrices between frames (see Figure 2). Let us say the correct transformation matrix T^1_{exact} (provided by Ground-Truth GPS data for instance) is obtained by a matrix product, which is mathematically denoted by (1).

$$T_{metrics} \Longrightarrow T^{ICP}_{exact} = \left(\boldsymbol{T}^2_{ICP}\right)^{-1} \cdot \left(T^1_2\right)^{-1} \cdot T^1_{exact}, \quad (1)$$

where $\boldsymbol{T}^2_{ICP}$ is the *ICP alignment* transformation matrix and T^1_2 is the initial transformation estimate (provided by GAA low lost GPS data for instance). T^{ICP}_{exact} will represent the transformation matrix used as metric to evaluate the refinement. That matrix T^{ICP}_{exact} will be denoted $T_{metrics}$.

Very often, alignments are evaluated using Euler representation of 6 parameters. The matrix elements ε_x, ε_y and ε_z could be used for the translation evaluation in the x, y, z axis; and ε_{roll}, ε_{pitch} and ε_{yaw} for the orientation. However in order to reduce the complexity of alignment analyse, we propose two indicators: ε^*_t and ε^*_r. We can can have a better idea of the orientation evaluation by re-expressing $T_{metrics}$ in terms of an Axis-Angle representation $(\theta, \mathbf{u})$ as is shown in (2). In the same way, translation evaluation can be reduced to analyse only the module of the vector with components ε_x, ε_y and ε_z (see (3)).

$$T_{metrics} = \begin{bmatrix} R_{3\times3} & \begin{matrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\end{matrix} \\ 000 & 1 \end{bmatrix}, R_{3\times3} \rightarrow (\theta, \mathbf{u}) \quad (2)$$

$$\varepsilon^*_t = \sqrt{(\varepsilon^2_x + \varepsilon^2_y + \varepsilon^2_z)} \quad , \quad \varepsilon^*_r = \theta, \quad (3)$$

TABLE I
INDICATORS: ICP ALIGNMENT EVALUATION IN SITUATIONS OF KNOWN MAPS TRANSFORMATION

Test N°	Initial Trasformation						Pre ICP		Post ICP		
	Translation (*m*)			Rotation (*rad*)			Metrics		Metrics		
	x	*y*	*z*	*roll*	*pitch*	*yaw*	ε_t	ε_r	ε^*_t	ε^*_r	*Aligned*
1	0	0	0	0	0	-0.07	0	0.715398	1.024417	**0.715846**	✗
2	0	0	0	0	0	-0.1		0.685398	1.029718	**0.692065**	✗
3	0	0	0	0	0	-0.5	0	0.285398	1.278808	**0.255834**	✗
4	0	0	0	0	0	- π/4	0	0	0	0	✓
5	0	0	0	0	0	-0.8	0	0.014602	0.047837	0.015796	✓
6	0	0	0	0	0	-1.5	0	0.714602	0.921425	**0.71481**	✗
7	0	0	0	0	0	-1.7	0	0.914602	0.837746	**0.896018**	✗
8	1	0	0	0	0	- π/4	1	0	0.449783	0.001196	✓
9	0	1	0	0	0	- π/4	1	0	0.508038	0.000691	✓
10	1	1	1	0	0	-0.6	1.732051	0.185398	0.157123	0.001292	✓
11	1	1	1	0	0	-0.8	1.732051	0.0146	0.470653	0.00249	✓
12	0	0	0	-0.3	0	- π/4	0	0.3	0.075291	0.423436	✓
13	0	0	0	0	-0.3	- π/4	0	0.3	0.072328	0.079704	✓
14	0	0	0	-0.3	-0.3	- π/4	0	0.423465	0.061049	0.592806	✓
15	1	1	0	-0.3	0	- π/4	1.414214	0.3	0.501968	0.423533	✓
16	0	0	1	-0.3	0	- π/4	1	0.3	0.33984	0.422476	✓
17	1	1	0	0	-0.3	- π/4	1.414214	0.3	0.548776	0.42313	✓
18	0	0	1	0	-0.3	- π/4	1	0.3	0.346891	0.423299	✓
19	3	0	0	0	0	- π/4	3	0	0.230664	0	✓
20	0	3	0	0	0	- π/4	3	0	0.290519	0	✓
21	5	0	0	0	0	- π/4	5	0	0.579472	0.001619	✓
22	0	5	0	0	0	- π/4	5	0	0.733493	0.000488	✓
23	5	5	0	0	0	- π/4	7.071069	0	0.831114	0.001691	✓
24	7	0	0	0	0	- π/4	7	0	0.000013	0	✓
25	0	7	0	0	0	- π/4	7	0	**6.252543**	0.004085	✗
26	7	7	0	0	0	- π/4	9.899497	0	**6.142731**	0.004367	✗
27	9	0	0	0	0	- π/4	9	0	0.437522	0	✓
28	0	9	0	0	0	- π/4	9	0	**8.470773**	0.004285	✗
29	9	9	0	0	0	- π/4	12.727925	0	**8.499197**	0.004931	✗

Table I shows the ICP alignment evaluation considering a known initial transformation. We also made the comparison between metrics before and after ICP. Analogously, in order to obtain indicators for correct ICP refinement, it was determined that values of $\varepsilon^*_r \leqslant 0.25$ and $\varepsilon^*_t \leqslant 6.14$ are considered as coherent merging results. In the real experiments where the exact correct transformation is not known, we can use the ICP matrix as a criterion of the error evaluation, in other words: $T_{metrics} = T^2_{ICP}$. In the following sections, our metrics from (3) will be renamed as $\bar{\varepsilon_r}^*$ and $\bar{\varepsilon_t}^*$, associated to experiments with unknown maps transformations.

III. METHODOLOGY

A. Pre-Local Mapping Stage

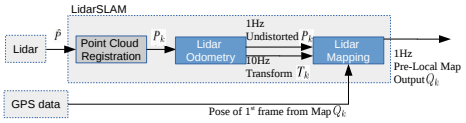


Fig. 3. Architecture of Pre-Local Mapping Stage

In the Pre-Local Mapping stage, inspired by the work in [11], its versatile configuration does not depend on a specific LidarSLAM method. A modified version of [1] [10] was used as LidarSLAM method for this article, considering its good place in the KITTI ranking [2]. Each mobile platform runs

[1]LOAM: `https://github.com/laboshinl/loam_velodyne`
[2]KITTI ranking: `http://www.cvlibs.net/datasets/kitti/eval_odometry.php`

this Pre-Local Mapping node. The behaviour of this node is illustrated in Figure 3, where $\hat{P}$ represents the raw point cloud obtained by a laser scan initially. The accumulated cloud during each sweep k generates P_k, which is processed by the *Lidar-Odometry* algorithm at a frequency around 10Hz. The algorithm receives P_k and computes the lidar motion between two successive sweeps, obtaining the transformation T_k. Deformation in P_k is corrected using the estimated lidar motion in order to use it at a frequency of 1Hz by the *Lidar-Mapping* algorithm. This algorithm executes the matching and registration of the non-distorted cloud onto a map. Both algorithms (Lidar-Odometry and Lidar-Mapping) are solved with a non-linear optimization, specifically the Levenberg-Marquardt method [13]. Finally, using the GPS information for identifying the initial vehicle pose, it is possible to coarsely project the map of each robot into a common coordinate frame. This projected cloud is denoted as the Pre-Local map. This Pre-Local mapping node can work with different kind of GPS data, either a Ground-Truth option (DGPS-RTK) or an approach most economical, for example using GPS-GGA type information (*Global Positioning System Fix Data*).

B. Local Mapping Stage

The architecture of Local Mapping Stage is described in Figure 4, where the process run on a robot referenced as "i", which shared its map with a robot "n".

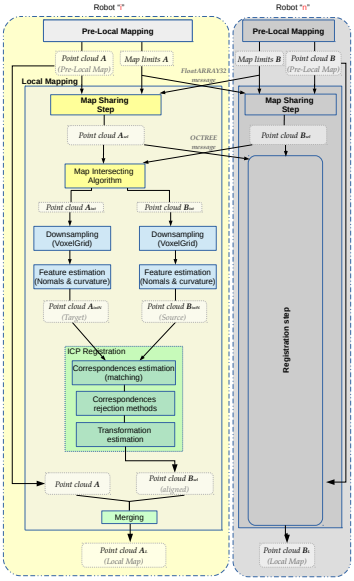


Fig. 4. Architecture of Local Mapping Stage for one robot "i", receiving map data from another robot "n".

1) Map Sharing Step: Here lies the key of the efficiency of the collaborative framework [11]. Once the Pre-Local Maps are generated, the robots have to exchange their point-clouds to start the map merging task. However in many cases, the sharing and processing of big map data can penalize the team performance with respect to memory usage and execution time. The sharing technique overcomes this problem, since each mobile platform just transfers a certain volume of its Pre-Local map to the other mobile units. Before the transfer of the map Previous the transference, a map compression step is executed in each vehicle, using octree structure from the octomap package [12] as a format of compression, optimizing the car-communication.

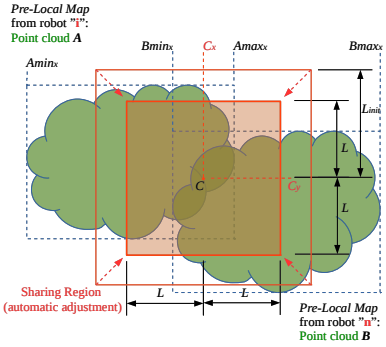


Fig. 5. Graphical representation of the Map Sharing technique (Top view of plane *XY*). $Amin_x$, $Amax_x$, $Bmin_x$ and $Bmax_x$ represent the point cloud limits along the x-axis.

An overview of this sharing technique is showed in Figure 5, in which the point clouds *A* and *B* are the Pre-Local Maps from two different cars "i" and "n" respectively. Both cars receiving simple data about the 3D limits of the maps (i.e. bounding cubic lattice of the point clouds) and the sharing node determines what part of its map will be sent to the other car. These limits are expressed as two vectors: in the first one $Amin$, the components represent the lowest displacement from the origin along each axis in the point cloud; and in the second one, $Amax$, is related to the point of highest displacement.

Data: Point Cloud A; Limits: Vectors $Amin$, $Amax$, $Bmin$ and $Bmax$; Parameters: Scalars L_{init}, L_{step}, Np_{max}
Result: Point Cloud A_{sel}
begin
  $A_{sel} \leftarrow \emptyset$;
  $C_x = 0$; $C_y = 0$; $C_z = 0$;
  $(V2_x, V3_x) =$
  $GetValues(Amin_x, Amax_x, Bmin_x, Bmax_x)$;
  $(V2_y, V3_y) =$
  $GetValues(Amin_y, Amax_y, Bmin_y, Bmax_y)$;
  $(V2_z, V3_z) =$
  $GetValues(Amin_z, Amax_z, Bmin_z, Bmax_z)$;
  $C_x = (V2_x + V3_x)/2$;
  $C_y = (V2_y + V3_y)/2$;
  $C_z = (V2_z + V3_z)/2$;
  $Np = PointSize(A)$;
  for $(L{=}L_{init}\ ;\ Np > Np_{max}\ ;\ L = L - L_{step})$ **do**
    $Smin_x = C_x - L$; $Smax_x = C_x + L$;
    $Smin_y = C_y - L$; $Smax_y = C_y + L$;
    $Smin_z = C_z - L$; $Smax_z = C_z + L$;
    foreach $\boldsymbol{a} \in A$ **do** ;
    if $Smin_x < a_x < Smax_x$ *and* $Smin_y < a_y < Smax_y$ *and* $Smin_z < a_z < Smax_z$ **then**
      $A_{sel} = A_{sel} + \boldsymbol{a}$;
    end
    $Np = PointSize(A_{sel})$;
  end
end

Algorithm 1: Selection of point cloud to share with another robot.

The pseudo-code of the map sharing step is depicted in Algorithm 1. First of all, the functions as $GetValues()$ sort in ascending order the array of components along each axis of the vectors $Amin$, $Amax$, $Bmin$, $Bmax$ and returns the 2nd and 3rd values from this sorted array, named $(V2)$ and $(V3)$ respectively. Then for each axis, the mean of $(V2)$ and $(V3)$ establishes the Cartesian coordinates (C_x,C_y,C_z)

of the geometric center of the exchanged region (S). This region S is a cube whose edge length is $2L$ and the points from A contained in this region are extracted to generate a new point cloud A_{sel}. At each iteration, the cube region is adjusted until the number of points from A_{sel} is less than a defined threshold (maximum number of points desired to exchange Np_{max}). Once the condition is accomplished the transfer begins. All the sharing process is analogous on the other robot "n". Finally, encoding of A_{sel} and B_{sel} in octree format is done to reduce the usage of bandwidth resources of the network.

2) Registration step with ICP: The intersecting volumes of A_{sel} and B_{sel} are calculated and renamed as A_{int} and B_{int}, which are down-sampled in order to reduce the time execution of the registration. In the feature descriptors estimation step, the surface normals and curvature of these input clouds are computed in order to improve the feature points matching, which is the most costly step of the ICP [14] [15]. Normal-pointclouds A_{intN} and B_{intN} generated after descriptors estimation are aligned with the ICP algorithm. The ICP refines an coarse alignment between clouds, estimating the best transformation to align a source cloud B_{intN} to a target cloud A_{intN} by iterative minimization of a cost function. Corresponding pairs (***b'***, ***a'***) from A_{intN} and B_{intN} are determined iteratively. Least squares registration is executed and the mean squared distance E is minimized with respect to translation $\boldsymbol{t}$ and rotation R:

$$E(R,\boldsymbol{t}) = \frac{1}{Np_{\boldsymbol{b'}}} \sum_{i=1}^{Np_{\boldsymbol{b'}}} \| \boldsymbol{a'}_i - (R\,\boldsymbol{b'}_i + \boldsymbol{t}) \|^2, \qquad (4)$$

where $Np_{b'}$ is the number of points ***b'***.

The resultant rotation matrix R and translation vector $\boldsymbol{t}$ are applied to source cloud B_{intN}. Again the matching between points from A_{intN} and B_{intN} is re-computed, until the variation of the mean square error between two consecutive iterations is less than a threshold. The final ICP refinement for n iterations is obtained by multiplying the individual transformations: $T_{ICP} = \prod_{j=1}^{n} T_j$. T_{ICP} is applied to B_{sel} to align and merge with the original cloud A, generating finally the Local Map A_L. Similarly, this merging process is executed individually in the other mobile unit.

IV. Results

Our experiments considered 3 vehicles: a *Renault ZOE*, a *Renault FLUENCE* and a *GOLFCAR* equipped with a *Velodyne* VLP-16 3D LiDAR of 360° horizontal and 30° vertical field of view.

Fig. 6. ZOE (left), FLUENCE (center) and GOLFCAR(right).

A. Urban scenarios: ECN campus case

1) For one robot: First of all, we analysed the impact of GPS quality on the 3D map generation of each vehicle. For that, we took as reference some tests executed with the FLUENCE car in Table I and we performed randomly a path with the robot around the surroundings of the ECN campus. The generated map was twice re-projected on a global frame using two types of GPS data: a high-accuracy DGPS-RTK and standard low cost GPS-GGA. The map with the 1st type of GPS was assumed as referential map, using this DGPS-RTK data as correct transformation matrix T^1_{exact}. On the other hand, the map re-located with GPS-GGA information was considered as a map with a coarsely placement, used to fill the initial transformation T^1_2. Results of ICP alignment between those 2 maps were addressed and substituted in (1), to obtain our metrics proposed in (3)

TABLE II
ALIGNEMNT EVALUATION: OUR INDICATORS VS BENCHMARK[2]

Robot	ε^*_t	ε^*_r	ε^*_t [2]	ε^*_r [2]
FLUENCE	1.781047	0.003453	3.005938	0.013086

The first two values in Table II corresponds to translation and orientation evaluation. Regarding the analysis performed in Table I, we have coherent alignment results using GGA vs Ground-Truth data because the values comply with the conditions $\varepsilon^*_t \leqslant 6.14$ m and $\varepsilon^*_r \leqslant 0.25$ rad. For the moment, we only show our indicator results with indicators proposed by [2], corresponding to the 3rd and 4th values in Table II. According to [2], coherent alignments must be $\varepsilon^*_t \leqslant 0.9$ and $\varepsilon^*_r \leqslant 0.21$, so their metrics showed only correct alignments in rotation.

2) For multi-robot team: In that case, our proposed system was validated considering two vehicles: FLUENCE and ZOE car. For both robots, ENU (East-North-Up) coordinate system was considered as external reference of the world frame $\{W\}$, where y-axis and x-axis correspond to the North and East respectively, but coinciding its origin with the GPS coordinates [Longitude: -1.547963; Latitude: 47.250229].

Experiments were realized in urban outdoor environment for an area of approximately 1000m x 700m. For the validation, the robots build clouds from different paths (see Figure 7), running pre-local mapping process in real-time.

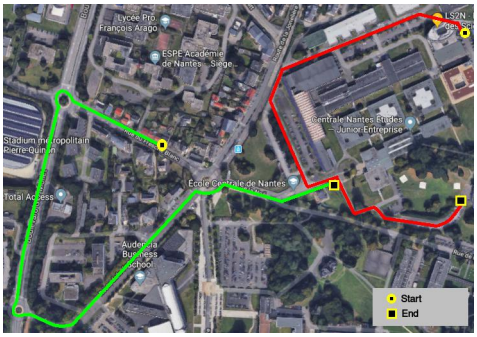


Fig. 7. Paths followed by the FLUENCE (green one) and the ZOE (red one) during experiments. Image source: Google Earth.

Results of the Pre-Local Mapping of this experiment are shown in Figure 8, which also illustrates the "sharing region" determined during the map exchange process in each robot. Pre-Local maps from the FLUENCE and ZOE cars were

projected on the global frame using Ground-Truth and GGA GPS data respectively. In order to simulate constraints on the network bandwidth and memory usage of the robot-team, the maximum number of points exchanged between cars was set to 410000.

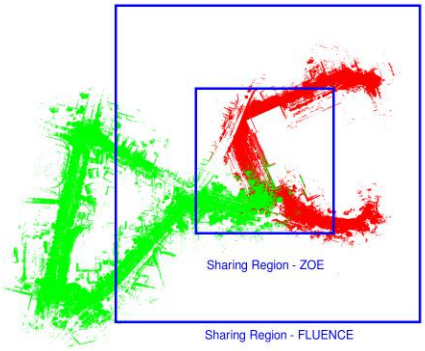
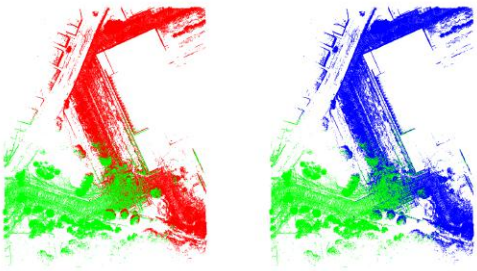


Fig. 8. Top view of unaligned Pre-Local Maps generated by FLUENCE (green one), and ZOE (red one) projected on a common coordinates system.

In the decentralized case, a meeting point for the robots was defined so they can transfer their maps and perform individually the relative registration process assuming its Pre-Local map as target cloud for alignment reference. Each mobile unit runs an intersecting algorithm, then an ICP refinement obtains an improved transform between maps. Figure 9 depicts the intersection between the shared point clouds during the ICP-alignment process for each robot.



(a) (b)

Fig. 9. Urban case: Alignment of the intersecting regions with ICP refinement performed in FLUENCE vehicle, when it received the ZOE map (a) Green and red maps represent the target and source clouds pre ICP, top view (b) Green and blue maps represent the target and aligned source clouds post ICP, top view.

TABLE III
RELATIVE ICP TRANSFORMATION FOR FLUENCE CAR

Cars	x	y	z	roll	pitch	yaw
F-Z	0.62208	-1.07885	5.03433	-0.00907	0.00294	-0.00232

$\bar{\varepsilon_r}^*$ = 0.009814 and $\bar{\varepsilon_t}^*$ = 5.186073 (Our indicators)
$\bar{\varepsilon_r}^*$ = 3.028573 and $\bar{\varepsilon_t}^*$ = 6.735261 (Jessup's indicators [2])

Quantitative refinement results are shown in Tables III and IV. ICP transformations were converted in an Euler parametrization $(x, y, z, roll, pitch, yaw)$ in meters and radians. Table III corresponds to the refinement process for the FLUENCE car, when it received the map from ZOE, and that map is aligned to the own pre-local map from FLUENCE.

TABLE IV
RELATIVE ICP TRANSFORMATION FOR ZOE CAR

Car	x	y	z	roll	pitch	yaw
Z-F	0.27346	0.48724	-6.05394	0.00346	0.00691	-0.00145

$\bar{\varepsilon_r}^*$ = 0.007873 and $\bar{\varepsilon_t}^*$ = 6.079665 (Our indicators)
$\bar{\varepsilon_r}^*$ = 3.023563 and $\bar{\varepsilon_t}^*$ = 6.814636 (Jessup's indicators [2])

We can also see either in Table III or Table IV that our proposed indicators confirm coherent merging results because the values are in the range $\varepsilon_r^* \leqslant 0.25$ and $\varepsilon_t^* \leqslant 6.14$ established from Table I. On the other hand, indicators from [2] do not work for this application because their results are not even in the ranges of translation ($\varepsilon_t^* \leqslant 0.9$) or rotation ($\varepsilon_r^* \leqslant 0.21$) established by themselves.

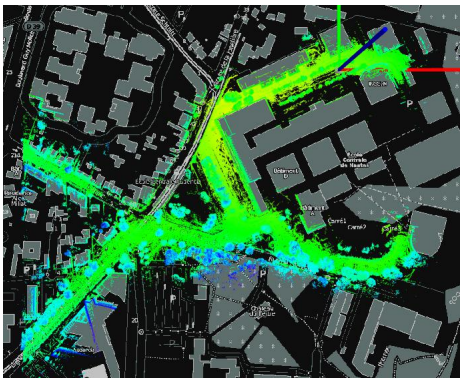


Fig. 10. Top view of final 3D Local Map of ZOE car.

Finally, Figure 10 depicts one of the final merging results, specifically the final 3D Local map from the ZOE projected on a 2D map in order to make qualitative comparisons.

B. Rural scenarios: farm case

For that case, our proposed system was validated using the GOLFCAR robot, but in order to do a collaborative mapping approach, the robot path was split in two for simulating two mobile units (see Figure 11(a)). All data come from the rural outdoor environment in an area of approximately 300m x 150m. The external reference of that world frame was also parallel to ENU coordinate system, with a GPS origin given [Longitude: -1.355357; Latitude: 46.809106].

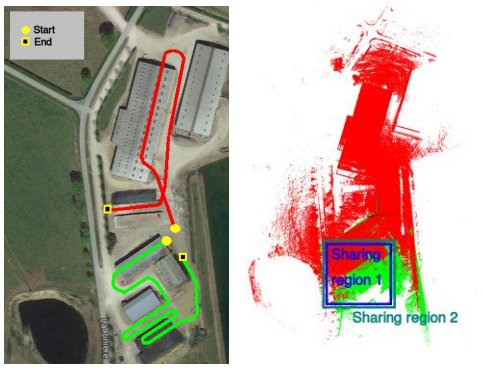
Initial Pre-Local maps and "sharing region" for each simulated robot are exposed in Figure 11(b). Those maps were projected on the global frame using GGA GPS data in both cases. For this scenario, the maximum number of points was set to 50000. Similar to the urban case, each platform mobile executes the intersecting algorithm and then an ICP alignment (see Figure 12)

TABLE V
RELATIVE ICP TRANSFORMATION FOR GOLFCAR(1)

Cars	x	y	z	roll	pitch	yaw
G1-G2	0.1461	0.3570	0.5108	-0.01526	-0.04404	0.00138

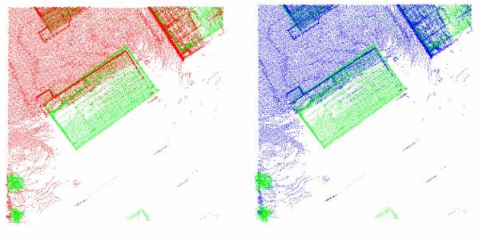
$\bar{\varepsilon_r}^*$ = 0.046647 and $\bar{\varepsilon_t}^*$ = 0.640166 (Our indicators)
$\bar{\varepsilon_r}^*$ = 3.119865 and $\bar{\varepsilon_t}^*$ = 1.014078 (Jessup's indicators [2])

Regarding the ICP transformation, quantitative alignment results relatives to GOLFCAR1(G1) are shown in Table V. All the ICP transformations are also expressed in Euler



(a) (b)

Fig. 11. (a) Paths followed by GOLFCAR1 (green one), GOLFCAR2 (red one) during experiments. Image source: Google Earth. (b) Top view of unaligned Pre-Local Maps generated by GOLFCAR1 (green one), and GOLFCAR2 robot (red one) projected on a common coordinate system.



(a) (b)

Fig. 12. Rural case: Alignment of the intersecting regions with ICP refinement performed in GOLFCAR1, when it received the GOLFCAR2 map (a) Green and red maps represent the target and source clouds pre ICP, top view (b) Green and blue maps represent the target and aligned source clouds post ICP, top view.

representation. Table V exposes also ICP alignments results with the respective indicators. Again indicators from [2] are not applicable on this case; on the other hand our proposed indicators confirm coherent merging results since values accomplish the conditions for translation and rotation ($\varepsilon_t^* \leqslant 6.14$, $\varepsilon_r^* \leqslant 0.25$). Finally, Figure 13 shows the final 3D map reconstruction for this scenario.

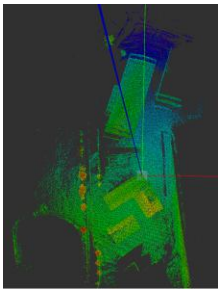


Fig. 13. Top view of final 3D Local Map for the rural environment

Tests demonstrated the influence of working with intersecting techniques, because it allows accelerating the registration by reducing the number of points for processing. In this context, experiments also showed the relevance of the map sharing algorithm by optimizing the performance of the robot-team by reducing the amount of data transferred on the network. Finally, the CoMapping framework remains an appropriate candidate to exchange and generate efficiently large maps with an approach multi-robot suitable for different kind of environments.

V. Conclusion and Future work

A cooperative framework for large scale maps applied to urban and rural environments was presented. In this framework, each robot generates its Pre-Local map using 3D-Lidar range measurements and executes individually the merging process considering initially a coarse map alignment with an optimum data exchange. The experimental results highlight the efficiency and versatility of the framework for cooperative mapping with three vehicles in different scenarios. Indicators were also proposed to demonstrate the success of the map merging process for the group of robots. Future work will be addressed to analyse the Sharing step and propose metrics to evaluate the map compression.

Acknowledgment

The authors gratefully acknowledge support of this work by Lionel Génevé and all the members of the ECN and LS2N institutions.

References

[1] P. Dinnissen, S. N. Givigi, and H. M. Schwartz, "Map merging of multi-robot slam using reinforcement learning." in *SMC*. IEEE, 2012.
[2] J. Jessup, S. N. Givigi, and A. Beaulieu, "Robust and Efficient Multirobot 3-D Mapping Merging With Octree-Based Occupancy Grids," *IEEE Systems Journal*, 2015.
[3] N. E. Özkucur and H. L. Akin, "Supervised feature type selection for topological mapping in indoor environments," in *21st Signal Processing and Communications Applications Conference, SIU 2013, Haspolat, Turkey, April 24-26, 2013*, 2013, pp. 1–4.
[4] J. Zhang and S. Singh, "Aerial and Ground-based Collaborative Mapping: An Experimental Study," in *The 11th Intl. Conf. on Field and Service Robotics (FSR)*, Sep 2017.
[5] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *In Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2003.
[6] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6D SLAM - 3d mapping outdoor environments: Research articles," *J. Field Robot.*, Aug. 2007.
[7] S. Kohlbrecher, O. V. Stryk, T. U. Darmstadt, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *in International Symposium on Safety, Security, and Rescue Robotics. IEEE*, 2011.
[8] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets - open-source library and experimental protocol." *Auton. Robots*, 2013.
[9] R. Zlot and M. Bosse, "Efficient Large-Scale 3D Mobile Mapping and Surface Reconstruction of an Underground Mine." in *FSR*, ser. Springer Tracts in Advanced Robotics, K. Yoshida and S. Tadokoro, Eds. Springer, 2012.
[10] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Robotics: Science and Systems*, 2014.
[11] L. Contreras, O. Kermorgant, and P. Martinet, "Efficient decentralized collaborative mapping for outdoor environments," in *2018 Second IEEE International Conference on Robotic Computing (IRC)*, 2018, pp. 56–63.
[12] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees," *Auton. Robots*, Apr. 2013.
[13] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.
[14] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, June 2001.
[15] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, Feb. 1992.