Active Collaborative Visual SLAM exploiting ORB Features

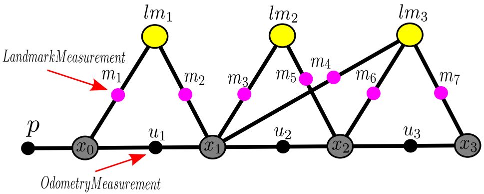
Muhammad Farhan Ahmed[1], Vincent Frémont[1], Isabelle Fantoni[2]

Ecole Centrale De Nantes (ECN)[1], Nantes, France
LS2N[2], CNRS, Nantes, France.

Outline

Simultaneous Localization And Mapping (SLAM)

1. Robot localizes itself and simultaneously maps the environment while navigating through it.
2. Localization is a problem of estimating the pose of the robot with respect to the map, while mapping makes up the reconstruction of the environment.
3. Modern SLAM approaches adopt a graphical approach. Where each node represents the robot or landmark pose and each edge represents a pose to pose or pose to landmark measurement measurement.
4. The objective of the SLAM problem is to find the optimal state vector x^* which minimizes the measurement error



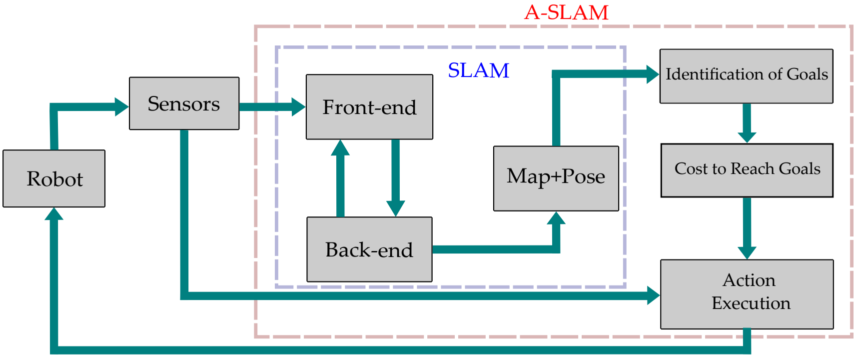
$$\mathbf{x}_i = \begin{pmatrix} x_i \\ y_i \\ \theta_i \end{pmatrix} \quad \mathbf{x}_l = \begin{pmatrix} x_l \\ y_l \end{pmatrix}$$

$$\mathbf{e}_i(\mathbf{x}) = \mathbf{Z}_i - f_i(\mathbf{x})$$
$$e_i(\mathbf{x}) = \mathbf{e}_i^T(\mathbf{x})\Omega_i \mathbf{e}_i(\mathbf{x})$$

$$\begin{aligned} \mathbf{x}^* &= \arg\min_x \textstyle\sum_i e_i(\mathbf{x}) \\ &= \arg\min_x \textstyle\sum_i \mathbf{e}_i^T(\mathbf{x})\Omega_i \mathbf{e}_i \mathbf{x} \end{aligned}$$

What is the Active SLAM (A-SLAM) problem?

1. A-SLAM deals with designing robot trajectories towards the goal locations subject to minimizing the uncertainty in its map localization.
2. The aim is to perform autonomous navigation and exploration of the environment without an external controller or human effort.



Proposed approach

- Motivated by our previous work[1]

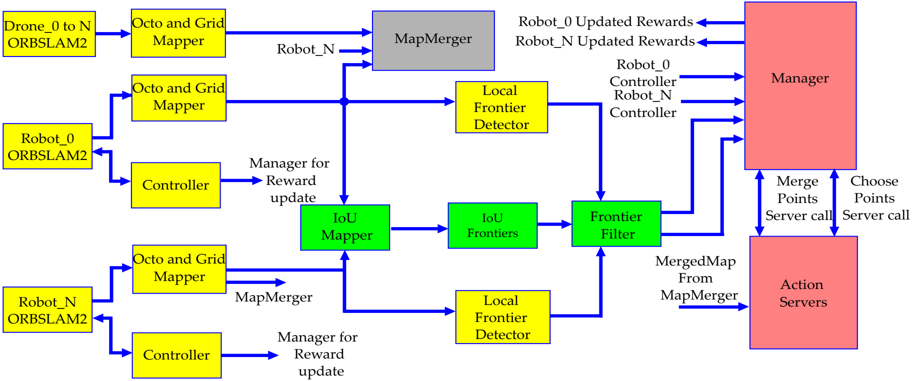


Figure 1: Central server, and local nodes communication.

[1] M. F. Ahmed, M. Maragliano, V. Frémont, C. T. Recchiuto and A. Sgorbissa"Efficient Frontier Management for Collaborative Active SLAM", MFI, 2024

Proposed approach

Algorithm 1: Compute IoU

Input: $M1$, $M2$
Output: $result.map$

$w, h \leftarrow$ width and height of IoU region **forall** h *and* w **do**
    $wx, wy \leftarrow$ grid to world coord.
    $idx1, idx2 \leftarrow$ world coord. to grid index
    $idx \leftarrow$ starting index for result.map **if** $[idx1]$ *and* $[idx2] \neq -1$ **then**
        if $[idx1] \wedge [idx2] = 0$ **then**
            $result.map[idx] \leftarrow 0$
        else if $[idx1] \wedge [idx2] = 100$ **then**
            $result.map[idx] \leftarrow 100$
        else if $[idx1] \vee [idx2] = 100$ **then**
            $result.map[idx] \leftarrow 100$
return $result.map$;

Algorithm 2: Frontier Filter

Input: M1_pts, IoU_pts, DIST_THRESH
Output: filtered_pts

$all\_pts \leftarrow$ M1_pts + IoU_pts;
$filtered\_pts \leftarrow \emptyset$;
forall p *in* all_pts **do**
    $too\_close \leftarrow$ False;
    forall fp *in* $filtered\_pts$ **do**
        if $dist(p, fp) < DIST\_THRESH$ **then**
            $too\_close \leftarrow$ True;
            break;
    if *not* too_close **then**
        add p to $filtered\_pts$;
return $filtered\_pts$;

Proposed approach

$$x^* = \arg\min_x \sum_i \mathbf{e}_i^T(x)\Omega_i \mathbf{e}_i(x) \tag{1}$$

$$\text{D-Opti} = \exp\left(\log\left(\det\left(\prod_{k=1,\dots l} \lambda_k\right)\right)\right)/n \tag{2}$$

Algorithm 3: Saved Goal Selection Based on Entropy

Input: SG_list, ORB_Stat, D-Opti, `D_MAX`, R_pos
Output: win_goal

if *(ORB_Stat is **lost**)* $\vee$ *(D-Opti $>$ `D_MAX`)* **then**
  foreach *item in SG_list* **do**
    $ent \leftarrow \text{entropy}(item_x, item_y, R\_pos)$;
    $egoal\_list \leftarrow (1 - ent)$;
  $win_{x,y} \leftarrow$ Max. value in $egoal\_list$;
  send $win_{x,y}$ to robot;
$reloc \leftarrow reloc + 1$; SG_list $\leftarrow win_{x,y}$;

Simulation Environment

- ROS Noetic, Gazebo, and Ubuntu 20.04 on Intel Core i7®, with 32Gb RAM and Nvidia RTX 1000. GPU.

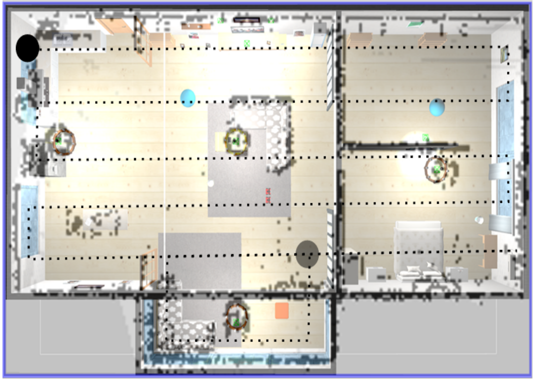


Figure 2: House Environment (H.E), 157M^2

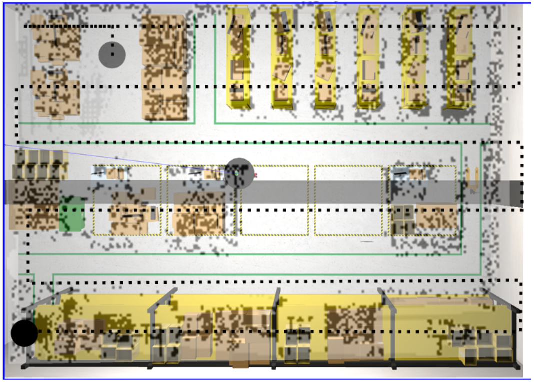
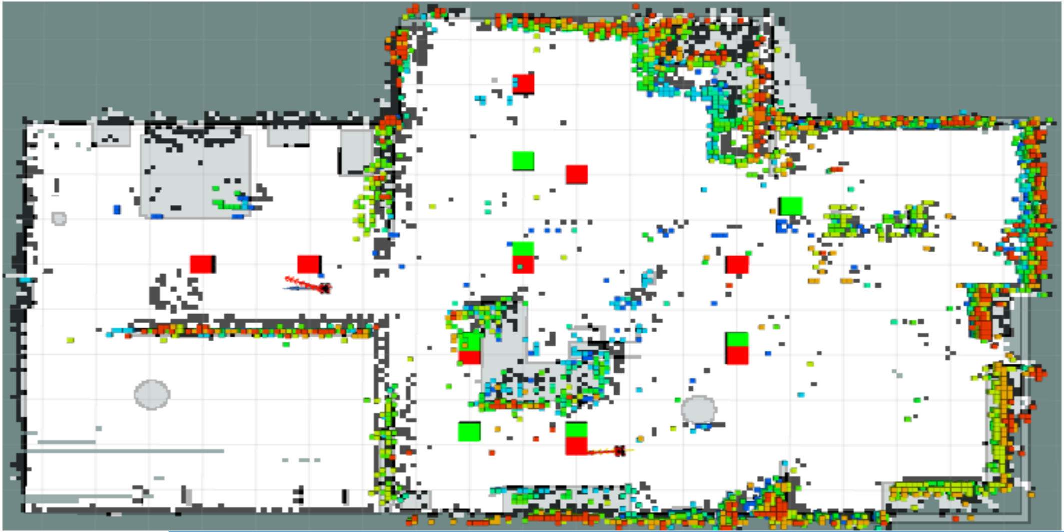
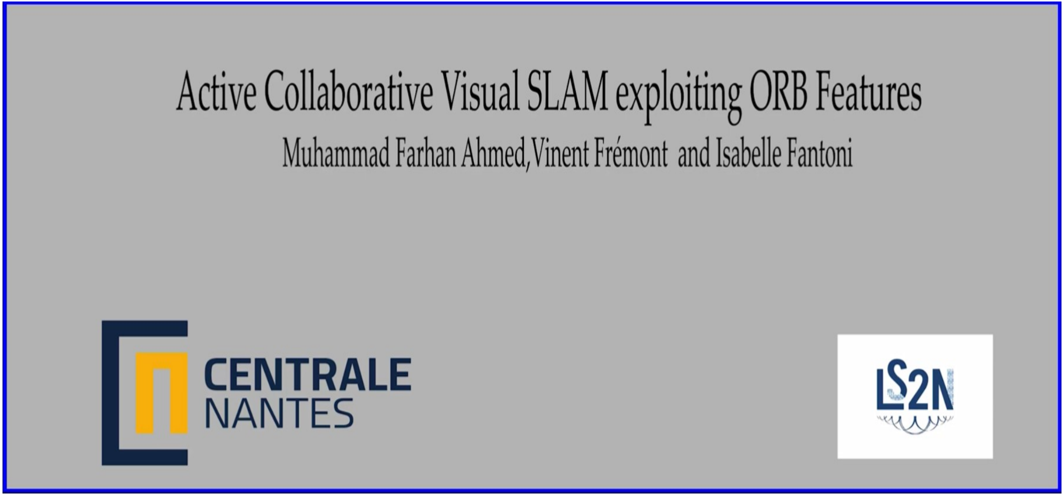


Figure 3: Warehouse Environment (W.E), 260M^2.

Simulation Environment



Simulation video



Simulation Results

- We conducted 10 simulations of 15 minutes each using 2 robots and 1 UAV for both H.G. and W.E. And compairing with MEXP[2] (red), DCM [3] (green), and Our (blue) methods rendering a total simulation time of 15 hours.

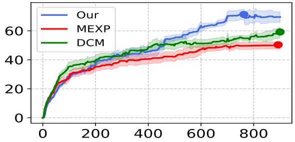


Figure 4: % area in H.G.

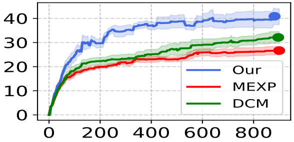


Figure 5: % of area in W.G.

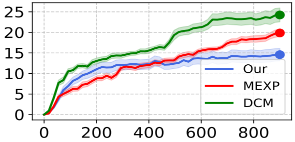


Figure 6: % of IoU area in H.E.

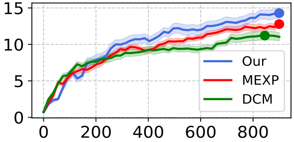


Figure 7: % of IoU area in W.G.

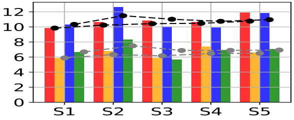


Figure 8: Filtered points in H.G.

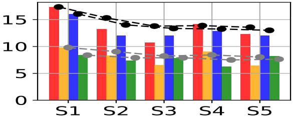
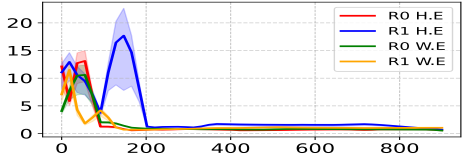


Figure 9: Filtered points in W.G.

[2] J. A. Placed et al."Explorb-slam: Active visual slam exploiting the pose-graph topology", ROBOT2022, 2023

[3] Anna B. et al "Decentralized strategy for cooperative multi-robot exploration and mapping", IFAC-PapersOnLine, 2020

Simulation Results



Robot	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
House Environment										
R0	4	0	3	2	0	0	2	0	0	0
R1	0	3	1	3	0	0	0	3	6	0
Total	4	3	4	5	0	0	2	3	6	0
Warehouse Environment										
R0	4	0	1	2	1	2	1	0	0	2
R1	1	2	0	0	1	3	2	0	0	0
Total	5	2	1	2	2	5	3	0	0	2

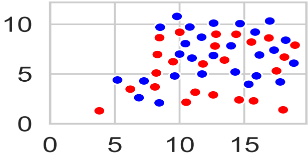


Figure 10: Goal points Our

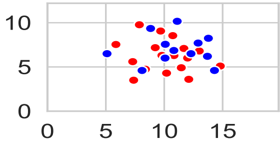


Figure 11: Goal points MEXP

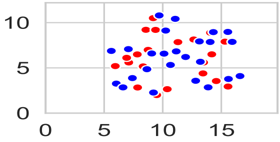


Figure 12: Goal points DCM

Env	Method	MSE	SSIM	NCC	CS
H.E	Our	**7544.238**	**0.386**	**0.426**	**0.455**
H.E	MEXP	7673.455	0.308	0.228	0.214
H.E	DCM	8695.255	0.198	0.125	0.155
W.E	Our	**8753.571**	**0.352**	**0.415**	**0.347**
W.E	MEXP	10160.746	0.265	0.344	0.262
W.E	DCM	11059.335	0.139	0.117	0.061

Conclusion

1. We proposed a method for the coordination of multiple robots in a collaborative exploration domain performing visual AC-SLAM.
2. We proposed a strategy to efficiently reduce the number of frontiers for the agents to compute their reward functions to reduce the computational cost and to spread the robots into the environment.
3. We also proposed a re-localization method to promote loop closure.
4. We presented extensive simulation analysis on publicly available environments and compared our approach to similar methods and achieved to explore an average of 32% and 27% more area.