Multi-robot navigation in cluttered and dynamic environment

Hugo Aguettaz
Nicolas Peslerbe
Marie-Joe Störi
Introduction - tasks

Goal:

- Avoid obstacle while keeping collective aggregation
- Maintain collective aggregation and cross another group of robots

Constraints:

- Minimize the use of dead reckoning because of the **noise additivity property**
- Minimize the communications between e-pucks because **very low communication rate** (32 bytes/s)
- Be aware of the **physical limitations of the sensors** (proximity sensors) and the **actuators** (motors)
Introduction - performance evaluation

- Orientation metric: Robots of a same group have to keep the same orientation

\[ o[t] = \frac{1}{N} \left| \sum_{n=1}^{N} e^{i \psi_n[t]} \right| \]

- Cluster metric: Robots of a same group have to keep close to each other

\[ c[t] = \left( 1 + \frac{1}{N} \sum_{n=1}^{N} \text{dist}(p_n[t], \bar{p}[t]) \right)^{-1} \]

- Velocity metric: Team has to go as fast as possible

\[ v[t] = \frac{1}{v_{max}} \max\{(\bar{p}[t] - \bar{p}_1[t-1])_\phi, 0\} \]
Experiments - Laplacian

- Instantaneous relative graph of a robot
  \[ \mathcal{G}_r(t) = (\mathcal{V}_r(t), \mathcal{E}_r(t)) \]

- Laplacian matrix of a graph
  \[ \mathcal{L} = \mathcal{I} \cdot \mathcal{W} \cdot \mathcal{I}^T \]

- Updated velocity
  \[ \dot{X}_{\mathcal{R}_r}(t) = \mathcal{L}(X_{\mathcal{R}_r}(t) - \mathcal{B}_{\mathcal{R}_r}(t)) \]
Experiments - Collective aggregation

- Add **attractive nodes** corresponding to robots in the graph
- Get relative positions using **Range and Bearing**
- Use **bias** to get the desired formation
Experiments - Obstacle avoidance

- Add virtual repulsive nodes corresponding to obstacles in the graph
- Get relative positions using proximity sensors
- No bias

Improvement: add virtual nodes at the back of the robot corresponding to fake obstacles with a certain life time when detected by left and right proximity sensors.
Experiments - Migratory urge

- Add **virtual attractive node** corresponding to the migratory urge
- Get relative positions using **dead reckoning** (realignment to the initial orientation)
- No bias
Experiments - Summary

\[ G_r(t) = (V_r(t), E_r(t)) \]
Experiments - Weights optimization

- **Optimization problem**
  1. Trial and errors
  2. P.S.O.

  - **Fitness function**
    \[
    F = V(1 - \Delta V)(1 - i) \left( \frac{\#\text{close robots}}{N} \right)
    \]
    - encourages the robot to go fast, go straight, avoid obstacles and be close to its friends.

  - **Weights of edges**
    - repulsive virtual nodes for the obstacle avoidance: 10 weights
    - attractive virtual node for the migration urge: 1 weight
    - weight of friend robots fixed: 1 weight
Experiments - P.S.O.

Technical details of the P.S.O.

- Homogeneous and public to achieve a common collaborative goal

Results and discussion about the P.S.O.

- Found solution performs well according to the fitness function ...
  - Go straight, avoid obstacles and be close to its friends
- ... but not according our initial requirements
  - Go through the arena: need to find another fitness function
Results - Formation init
Results - Static obstacles
Results - Dynamic obstacles
Results - Real E-puck
Conclusion

Works well on simulations but many difficulties:

- To deal with the non holonomicity of the e-puck
- To find the right hyper-parameters (threshold of proximity sensors, lifetime of fake obstacles, ...)

Hard to put on real e-pucks because of the limitations due to:

- The proximity sensors defaults
- The low rate communication between e-pucks
- The dead reckoning
- The range and bearing
- The computational power of e-pucks
Thank you for your attention

Do you have any questions?