Human-Guided Unmanned Surface Vehicle Teams

Pradeep Rajendran, Brual Shah, and Shaurya Shriyam, Petr Svec, and Satyandra K. Gupta
Motivation

• Many civilian applications can benefit from deploying a human-robot team consisting of several small USVs and one or more human supervisors
  – Reduces costs
  – Improves safety
  – Increases operational efficiencies

• Representative applications:
  – Remote/persistent ocean sensing
  – Marine search and rescue
  – Maritime operations in congested port environments
  – Industrial offshore supply and support
Planning Challenges

- Need to consider and utilize underlying physics
- Need to optimize planning time
- Need to proactively manage risk

https://youtu.be/ePDoCPI06rk?t=16s
Research Areas

Long distance path planning

Energy efficient path planning

On-Line Task Decomposition for Collaborative Surveillance

Risk-aware, COLREGs-compliant trajectory planning

Planning for target following with motion prediction

Dynamics-Aware Reactive Trajectory Planning
- Develop an optimal long distance path planner that uses A* on visibility graph defined over quadtrees
- Develop an path planning algorithm that uses the weather forecast and generate energy efficient paths in time-varying flow fields (e.g. ocean currents)
- Generate a meta-model of USV’s dynamics and PID / LQR / backstepping controller
- Develop a global, resolution-adaptive, dynamically-feasible, COLREGs-compliant, 5D trajectory planner
- Demonstrate capabilities of the planner by performing simulation and physical experiments
Task decomposition for collaborative surveillance by multiple USVs

Presented at Conference IDETC 2017
Deploying a team of robots instead of a single robot can improve safety, increase operational efficiencies, and significantly reduce costs.

Collaborative tasks involving one or more team(s) of robots require spatial partitioning of the region of interest.

Decomposition of tasks among team(s) of robots is a very challenging problem because:
- Imperfect information of the environment
- Changing environmental conditions
- Varying performance of the robot as a result of its interaction with the environment
- Risk of collision with external entities (e.g., civilian traffic)
A sample frontier based area expansion of a single USV is shown.
The cells lying inside region of interest show the values obtained from velocity map.
These values are used to explore the frontier cells in the next time step.
But our algorithm only focuses on generating equitable area partitions.
The coverage trajectories arising out of this area exploration technique is not outputted as a result.
Since all USVs expand their areas, the USV that finishes last is the bottleneck agent.

The time taken by the bottleneck USV is minimized using the particle Swarm Optimization algorithm.

The optimization variable is only considered to be the initial placement of the USVs along the region boundary.

Suppose USV $U_i$ finishes in time $t_{if}$. Then we optimize the time $t_{finish}$ taken by bottleneck agent $i^*$ given by:

\[
t_{finish} = \max_i(t_{if}) \quad \quad i^* = \arg \max_i(t_{if})
\]
Illustrative Examples
Long Distance Path Planner

Presented at Conference ICAPS 2016
Submitted in Journal Autonomous Robots
- USVs are increasingly being used on long missions with large operating space and time
- The real-world scenarios comprises of complex polygons and cannot be represented by standard geometrical shapes
- The available free space in marine environment changes over time:
  - Low tides may make it infeasible to go through shallow waters
  - Environmental restrictions may present the USVs to go through certain protected marine regions for certain periods
  - Weather induced waves may prohibit traveling over certain areas due to high collision risks
Related Work

### Techniques for Path Planning

<table>
<thead>
<tr>
<th>Techniques for Path Planning</th>
<th>Representative References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid-based approach</td>
<td>• A*: Hart, Nilsson, and Raphael 1968</td>
</tr>
<tr>
<td></td>
<td>• Block A*: Yap et. al. 2002</td>
</tr>
<tr>
<td></td>
<td>• Theta*: Daniel et. al. 2010</td>
</tr>
<tr>
<td>Sampling-based approach</td>
<td>• RRT: LaValle 2001, RRT*: S. Karaman 2010</td>
</tr>
<tr>
<td></td>
<td>• Roadmaps: Kavraki et al. 1996</td>
</tr>
<tr>
<td></td>
<td>• Roadmaps and Voronoi diagrams: Bhattacharya et. al. 2008</td>
</tr>
<tr>
<td>Visibility-graph based approach</td>
<td>• Visibility graphs: Lee et. al. 1978 and Lozano-Perez et. al. 1979</td>
</tr>
<tr>
<td></td>
<td>• Visibility binary trees: Rashid et. al. 2013</td>
</tr>
</tbody>
</table>

- Grid-based methods with constant grid size are inefficient and computationally expensive. Optimality of the path depends upon the resolution of the grid.
- Sampling-based methods produce non-optimal paths and the computation time is dependent on map specific parameter like $d_{min}$ (in RRT*).
- Visibility graphs are grid independent and computes optimal paths but computationally inefficient for online planning.
Overview of Approach

Path Planner

- Computation of VG using Quadtree
- Efficient computation of VG
- Heuristic for A* Search
- Focusing the A* Search

VG: Visibility Graph
Background on Path Planning using Visibility Graph

- Represent the polygons imported from the nautical charts into quadtrees
- Options to compute visibility graph:
  - Precompute the visibility graph and use Dijkstra's to compute shortest path
    - High computational time \(O(n^2 \log(n))\)
  - Compute the visibility graph on the fly and search using A*
    - Computational time can still be high in challenging scenarios (worst case \(O(n^2 \log(n))\))
- The complexity of VG is reduced by only considering the tangents to the obstacles without compromising the optimality
- The reduced VG is termed as Tangent Graph (TG)

The number of nodes in visibility graphs are large and makes the search inefficient.

Optimal path will not pass through vertices that lies in the globally concave region.

Elimination of globally concave nodes:
- Compute the island regions by combining the connected component of solid leaf nodes of quadtree.
- Compute the non-intersecting convex hulls for each island.
- Eliminate the nodes that lie in the interior of the convex hulls.

The path is generated by computing the visible nodes from each expanded node during the search.

The next node is selected based on least-cost A* fashion.
Computational Results

- TG: Tangent Graph
- RTG: Reduced Tangent Graph
- Approximately 70-80% of the nodes from the TG are eliminated

<table>
<thead>
<tr>
<th># of Quad Nodes</th>
<th># of Nodes in TG</th>
<th># of Nodes in RTG</th>
<th>% reduction in number of nodes in RTG over TG</th>
</tr>
</thead>
<tbody>
<tr>
<td>6446</td>
<td>6132</td>
<td>919</td>
<td>85.01</td>
</tr>
<tr>
<td>6578</td>
<td>6272</td>
<td>1272</td>
<td>79.72</td>
</tr>
<tr>
<td>7622</td>
<td>7832</td>
<td>1044</td>
<td>86.67</td>
</tr>
<tr>
<td>8689</td>
<td>8668</td>
<td>823</td>
<td>90.51</td>
</tr>
<tr>
<td>20605</td>
<td>17548</td>
<td>1865</td>
<td>89.37</td>
</tr>
<tr>
<td>21075</td>
<td>18268</td>
<td>2678</td>
<td>85.34</td>
</tr>
<tr>
<td>24951</td>
<td>21428</td>
<td>3255</td>
<td>84.81</td>
</tr>
<tr>
<td>23567</td>
<td>19896</td>
<td>1832</td>
<td>90.79</td>
</tr>
<tr>
<td>50534</td>
<td>43340</td>
<td>9553</td>
<td>77.96</td>
</tr>
<tr>
<td>55180</td>
<td>42868</td>
<td>9785</td>
<td>77.17</td>
</tr>
<tr>
<td>54961</td>
<td>40668</td>
<td>11574</td>
<td>71.54</td>
</tr>
<tr>
<td>45082</td>
<td>36936</td>
<td>6652</td>
<td>81.99</td>
</tr>
<tr>
<td>75578</td>
<td>59344</td>
<td>22571</td>
<td>61.97</td>
</tr>
<tr>
<td>76507</td>
<td>47132</td>
<td>19837</td>
<td>57.91</td>
</tr>
<tr>
<td>75017</td>
<td>45316</td>
<td>20427</td>
<td>54.92</td>
</tr>
<tr>
<td>74375</td>
<td>46344</td>
<td>19340</td>
<td>58.27</td>
</tr>
<tr>
<td>96480</td>
<td>81568</td>
<td>14924</td>
<td>81.70</td>
</tr>
<tr>
<td>100657</td>
<td>85332</td>
<td>16144</td>
<td>81.08</td>
</tr>
<tr>
<td>96630</td>
<td>82032</td>
<td>14422</td>
<td>82.42</td>
</tr>
<tr>
<td>101056</td>
<td>85616</td>
<td>17055</td>
<td>80.08</td>
</tr>
</tbody>
</table>
Low computational efficiency of A* on standard tangent graphs is due to:
- Large branching factor of A* search
- Performing collision checks to determine the visible nodes in the graph

The number of collision checks is reduced by eliminating the nodes that lie in the shadow region.
- Method 1 (M1): Uses the tangent graph (TG) with Euclidean distance to goal as heuristic cost (ECU)

- Method 2 (M2): Uses the reduced tangent graph (RTG) with Euclidean distance (ECU) and Reduced Collision Checks (RCC)

- Method 2 reduces the computation time by approximately 40% over Method 1

<table>
<thead>
<tr>
<th># of Quad Nodes</th>
<th>Computation Time (in sec)</th>
<th>% reduction in computation time by M1 over M1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TG + ECU (M1)</td>
<td>RTG + ECU + RCC (M2)</td>
</tr>
<tr>
<td>6446</td>
<td>23.33</td>
<td>12.92</td>
</tr>
<tr>
<td>6578</td>
<td>19.59</td>
<td>13.40</td>
</tr>
<tr>
<td>7622</td>
<td>19.90</td>
<td>10.51</td>
</tr>
<tr>
<td>8689</td>
<td>17.95</td>
<td>8.99</td>
</tr>
<tr>
<td>20605</td>
<td>98.18</td>
<td>63.58</td>
</tr>
<tr>
<td>21075</td>
<td>184.79</td>
<td>81.42</td>
</tr>
<tr>
<td>24951</td>
<td>60.83</td>
<td>16.81</td>
</tr>
<tr>
<td>23567</td>
<td>133.16</td>
<td>59.33</td>
</tr>
<tr>
<td>50534</td>
<td>694.65</td>
<td>374.48</td>
</tr>
<tr>
<td>55180</td>
<td>857.49</td>
<td>590.52</td>
</tr>
<tr>
<td>54961</td>
<td>597.41</td>
<td>392.40</td>
</tr>
<tr>
<td>45082</td>
<td>204.37</td>
<td>114.45</td>
</tr>
<tr>
<td>75578</td>
<td>2050.11</td>
<td>1310.76</td>
</tr>
<tr>
<td>76507</td>
<td>1354.64</td>
<td>730.12</td>
</tr>
<tr>
<td>75017</td>
<td>728.15</td>
<td>433.00</td>
</tr>
<tr>
<td>74375</td>
<td>874.43</td>
<td>535.61</td>
</tr>
<tr>
<td>96480</td>
<td>2796.71</td>
<td>1129.96</td>
</tr>
<tr>
<td>100657</td>
<td>1124.98</td>
<td>644.52</td>
</tr>
<tr>
<td>96630</td>
<td>984.71</td>
<td>480.56</td>
</tr>
<tr>
<td>101056</td>
<td>1492.80</td>
<td>682.06</td>
</tr>
</tbody>
</table>
- Optimal path towards the goal have to circumvent the largest obstacle that lie on the straight line path connecting the start and the goal
- The developed new heuristic avoids expanding nodes on the boundary of the obstacle and improves the efficiency of the A* search

Let's define the heuristic cost as follows:

\[ h_T^R(n) : \text{Heuristic cost of the path passing right side of the obstacle} \]
\[ h_T^L(n) : \text{Heuristic cost of the path passing left side of the obstacle} \]

\[ h_T^R(n) = d(n, v(\max(d_{R1}, d_{R2}))) + d(v(\max(d_{R1}, d_{R2})), n_G) \]

\[ h_T(n) = \min(h_T^L(n), h_T^R(n)) \]
Computational Results: New Heuristic

- Method 1 (M1): Uses the tangent graph (TG) with Euclidean distance to goal as heuristic cost (ECU)

- Method 3 (M3): Uses the reduced tangent graph (RTG) with the developed heuristic (HEU) and Reduced Collision Checks (RCC)

- Method 3 reduces the computation time by approximately 70% over Method 1

<table>
<thead>
<tr>
<th># of Quad Nodes</th>
<th>Computation Time (in sec)</th>
<th>% reduction in computation time by M1 over M3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TG + ECU (M1)</td>
<td>RTG + HEU + RCC (M3)</td>
</tr>
<tr>
<td>6446</td>
<td>23.33</td>
<td>3.24</td>
</tr>
<tr>
<td>6578</td>
<td>19.59</td>
<td>3.60</td>
</tr>
<tr>
<td>7622</td>
<td>19.90</td>
<td>2.65</td>
</tr>
<tr>
<td>8689</td>
<td>17.95</td>
<td>2.90</td>
</tr>
<tr>
<td>20605</td>
<td>98.18</td>
<td>14.94</td>
</tr>
<tr>
<td>21075</td>
<td>184.79</td>
<td>33.23</td>
</tr>
<tr>
<td>24951</td>
<td>60.83</td>
<td>7.41</td>
</tr>
<tr>
<td>23567</td>
<td>133.16</td>
<td>29.85</td>
</tr>
<tr>
<td>50534</td>
<td>694.65</td>
<td>157.65</td>
</tr>
<tr>
<td>55180</td>
<td>857.49</td>
<td>226.61</td>
</tr>
<tr>
<td>54961</td>
<td>597.41</td>
<td>245.68</td>
</tr>
<tr>
<td>45082</td>
<td>204.37</td>
<td>59.74</td>
</tr>
<tr>
<td>75578</td>
<td>2050.11</td>
<td>546.60</td>
</tr>
<tr>
<td>76507</td>
<td>1354.64</td>
<td>432.97</td>
</tr>
<tr>
<td>75017</td>
<td>728.15</td>
<td>235.56</td>
</tr>
<tr>
<td>74375</td>
<td>874.43</td>
<td>233.40</td>
</tr>
<tr>
<td>96480</td>
<td>2796.71</td>
<td>761.32</td>
</tr>
<tr>
<td>100657</td>
<td>1124.98</td>
<td>364.58</td>
</tr>
<tr>
<td>96630</td>
<td>984.71</td>
<td>360.96</td>
</tr>
<tr>
<td>101056</td>
<td>1492.80</td>
<td>421.08</td>
</tr>
</tbody>
</table>
- A* algorithm needs to explore all possible tangent nodes
- Large spatial regions, will have large number of tangent nodes that needs to examined
- In the scenario shown, the number of edges on the tangent graph are very large even the path to goal is easy to compute

Pathological scenario where a node in tangent graph has a large branching factor.
• Consider local nodes as candidates for visibility checks (i.e. nodes that lie within a circle of predetermined size)

• Bias the candidate nodes being considered towards the goal by selecting nodes on the convex hull which intersect the line from the current state to the destination state

• Significantly reduces the candidate tangent graph nodes to process for visibility checks
- Method 1 (M1): Uses the tangent graph (TG) with Euclidean distance to goal as heuristic cost (ECU)
- Method 4 (M4): Uses the focused search on reduced tangent graph (FS) with the developed heuristic (HEU) and Reduced Collision Checks (RCC)
- Method 4 reduces the computation time by approx. 90% over Method 1
- Computational performance comes at the cost of marginal loss of optimality and increased path length by approx. 1%

### Computational Results: Focused Search

<table>
<thead>
<tr>
<th># of Quad Nodes</th>
<th>Computation Time (in sec)</th>
<th>% reduction in computation time by M1 over M4</th>
<th>% increase in path length by M4</th>
</tr>
</thead>
<tbody>
<tr>
<td>6446</td>
<td>23.33</td>
<td>94.95</td>
<td>0.00</td>
</tr>
<tr>
<td>6578</td>
<td>19.59</td>
<td>95.78</td>
<td>0.00</td>
</tr>
<tr>
<td>7622</td>
<td>19.90</td>
<td>96.45</td>
<td>0.01</td>
</tr>
<tr>
<td>8689</td>
<td>17.95</td>
<td>97.15</td>
<td>0.00</td>
</tr>
<tr>
<td>20605</td>
<td>98.18</td>
<td>95.21</td>
<td>0.16</td>
</tr>
<tr>
<td>21075</td>
<td>184.79</td>
<td>97.88</td>
<td>0.13</td>
</tr>
<tr>
<td>24951</td>
<td>60.83</td>
<td>97.48</td>
<td>0.10</td>
</tr>
<tr>
<td>23567</td>
<td>133.16</td>
<td>96.49</td>
<td>0.26</td>
</tr>
<tr>
<td>50534</td>
<td>694.65</td>
<td>94.82</td>
<td>0.67</td>
</tr>
<tr>
<td>55180</td>
<td>857.49</td>
<td>91.12</td>
<td>0.58</td>
</tr>
<tr>
<td>54961</td>
<td>597.41</td>
<td>87.13</td>
<td>0.50</td>
</tr>
<tr>
<td>45082</td>
<td>204.37</td>
<td>93.21</td>
<td>0.83</td>
</tr>
<tr>
<td>75578</td>
<td>2050.11</td>
<td>91.76</td>
<td>0.97</td>
</tr>
<tr>
<td>76507</td>
<td>1354.64</td>
<td>88.08</td>
<td>1.13</td>
</tr>
<tr>
<td>75017</td>
<td>728.15</td>
<td>92.17</td>
<td>1.09</td>
</tr>
<tr>
<td>74375</td>
<td>874.43</td>
<td>91.23</td>
<td>1.08</td>
</tr>
<tr>
<td>96480</td>
<td>2796.71</td>
<td>93.91</td>
<td>1.90</td>
</tr>
<tr>
<td>100657</td>
<td>1124.98</td>
<td>80.85</td>
<td>1.33</td>
</tr>
<tr>
<td>96630</td>
<td>984.71</td>
<td>86.66</td>
<td>1.12</td>
</tr>
<tr>
<td>101056</td>
<td>1492.80</td>
<td>90.17</td>
<td>1.98</td>
</tr>
</tbody>
</table>
### Scaling of our Algorithm

<table>
<thead>
<tr>
<th>Map Size (pixels)</th>
<th>Computation Time (in sec)</th>
<th>Theta*</th>
<th>Our Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024 x 1024</td>
<td></td>
<td>1.87</td>
<td>0.19</td>
</tr>
<tr>
<td>2048 x 2048</td>
<td></td>
<td>16.69</td>
<td>0.22</td>
</tr>
<tr>
<td>4096 x 4096</td>
<td></td>
<td>90.784</td>
<td>0.24</td>
</tr>
</tbody>
</table>

- Our approach is scalable with the size of the operating environment as compared to Theta*
Results: Real Scenario

- Map size: 100 x 100 km
- # of Quadtree nodes: 66425
- Depth of the quadtree: 13 i.e. smallest resolution 12.207 m
- # of tangent graph nodes: 2732
- # of nodes expanded by focused search: 711
- Computation time by our algorithm is 12.19 sec
Assessing Effectiveness of Focused Search

- **Possibility #1:** Next node on the optimal path is in the local search region (LSR) \( n_{local}^{opt} \)

- **Possibility #2:** Next node on the optimal path (NNOP) is NOT in the local circular search region but lies on the island that intercepts direct line to the goal \( n_{ext}^{opt} \)

- **Possibility #3:** Next node on the optimal path (NNOP) is NOT in the local circular search region but back connection is able to find it \( n_{back}^{opt} \)

- **Possibility #4:** Next node on the optimal path cannot be found \((100 - n_{local}^{opt} - n_{ext}^{opt} - n_{back}^{opt})\)
Effect of the size of LSR on $n^{opt}_{local}$

- Curve increases in super-linear fashion and then begins to saturate as we reach higher percentage of the mean tangent length.

LSR: Local Search Region

$n^{opt}_{local}$: Next Node on Optimal Path that lies inside the local search region
Effect of the size of LSR on $n_{\text{ext}}^{\text{opt}}$

- Plot shows the percentage of remaining nodes on optimal path after eliminating the nodes that lie inside the local search region i.e. $100 - n_{\text{local}}^{\text{opt}}$
- Approximately 65% of the remaining nodes lie inside the extended search region

LSR: Local Search Region

$n_{\text{ext}}^{\text{opt}}$: Next Node on Optimal Path that lies inside the extended search region
Plot shows the percentage of remaining nodes on optimal path after eliminating the nodes that lie inside the local and extended search region i.e. $100 - n_{local}^{opt} - n_{ext}^{opt}$.

Significant amount of optimal path segment are larger than mean tangents length.
Effect of the size of LSR on $n_{opt}^{back}$

- Plot shows percentage of the next nodes on the optimal path do not lie in the local and extended search region and are not discovered by the back connection approach i.e. $100 - n_{local}^{opt} - n_{ext}^{opt} - n_{back}^{opt}$

- Selecting the size of LSR approximately one and half times the mean tangent length ensures that the probability of not finding the optimal path drop below 1%.

- For example at mean (i.e. 100%), the probability of not finding node = $(1-0.5)*(1-0.65)*(1-0.85) = 0.026 = 2.6\%$

LSR: Local Search Region
Handling Time Varying Free Space

Wait action executed just before the collision and the radius of circle is proportional to the wait time.
- Developed a resolution independent, any-angle path planning algorithm that computed paths on maps of 100 x 100 Km with occupancy of 40% in approximately 2 min (Python implementation)

- Computational performance is improved by approx. 90% as compared to tangent graph based approach with marginal loss of optimality

- Extended the planner to compute paths in an environment with time varying free space
Deliberative Trajectory Planner

Presented at Conference *IROS 2014*
Published in Journal *Autonomous Robots 2016*
Applications of underlying algorithm is published in Journal of Ocean Engineering 2016
Trajectory planning is a congested and highly dynamic environment with civilian traffic is a non-trivial planning problem.

- Need to consider vehicle’s non-linear dynamics and control in planning.
- Need to employ a short sense-plan-control cycle especially when operating at high speeds.
- Need to follow Coast Guard Collision Regulations (COLREGs).
- Needs to consider intentions of civilian vessels while performing avoidance maneuvers.
- Need to find a balanced trade-off among the trajectory length, collision risk, and violation of COLREGs.
Overview of Approach

Trajectory Planner

- Generation of Meta-model
- Intension Modelling of Civilian Vessel
- RCAP
- Congestion Metric
- A-RCAP

RCAP: Risk and Contingency Aware Planner
\[ X = (m - X\ddot{u})\dot{u} - (m - Y\dot{v})vr + Y\dot{r}r^2 - X\dot{u}u, \]
\[ Y = (m - Y\dot{v})\dot{v} - Y\dot{r}\dot{r} + (m - X\dot{u})ur - Y\dot{v}v - Y\dot{r}r, \]
\[ N = (I_z - N\dot{r})\dot{r} - Y\dot{r}\dot{v} + (X\dot{u} - Y\dot{v})uv - Y\dot{r}ur - N\dot{v}v - N\dot{r}r, \]
\[ X = T_p + T_s, \]
\[ Y = 0, \]
\[ N = (T_p - T_s)\frac{t}{2}. \]
Intention Modelling of Civilian Vessels

- **Uncertainty in Civilian Vessels:**
  - Mean: Lies on trajectory provided by intention model
  - Variance:

  $$
  \sum_{b_{l,t}} = \begin{bmatrix}
  c & -s \\
  s & c
  \end{bmatrix}^{-1}
  \begin{bmatrix}
  \alpha_{\Sigma,x} \sigma^2_x \\
  0 \\
  0 \\
  \alpha_{\Sigma,y} \sigma^2_y
  \end{bmatrix}
  \begin{bmatrix}
  c & -s \\
  s & c
  \end{bmatrix}
  $$

  where $c = \cos(\psi_{b_l})$ & $s = \sin(\psi_{b_l})$
Risk and Contingency-Aware Trajectory Planner (RCAP)

- Developed a 5D $[x, y, \psi, u, t]^T$ lattice-based global planner RCAP
- Incorporated the reciprocal behavior of civilian vessels into planning
- Reduced the collision rate by integrating the contingency maneuvers directly into the search
Adaptively scaled the control primitives to speed-up the search

A set of control action primitive divided into three regions
$\text{Congestion Metric}$

$\text{RIC}_r = \bigcup_{l=1, d_l < r}^L \text{RIC}_l$

$\Lambda_{cgn} = \int_{r=0}^{\infty} \frac{\text{RIC}_r}{A_r} \, dr$

$\text{RIC}_i : \text{Region of inevitable collision}$
Simulation Results

- 61-87% reduction in collision rate as compared to COLREGs-compliant VO-based planner
- Reduced the distance travelled by additional ~4% by computing globally optimal trajectories
Energy-efficient Path Planner

Presented at PlanRob Workshop held at ICAPS 2016
- Performance of the USVs are influenced by the underlying medium flow in which they operate
- Usually, USVs follow a shortest distance path by using a feedback controller to compensate for the disturbances, but that leads to:
  - Significant consumption of energy
  - Wear and tear of actuator over a long period of operation
- It will be advantageous for the USVs to exploit the medium flows instead of overcoming it, that provides:
  - Increase in overall mission duration
Overview of Approach

Path Planner

- Motion Model
- Admissible Heuristic for A* Search
- Start Time Optimization
- The motion primitive set consists of actuation action and a special free-flow action.
- Transition from the current state \( s \) to neighboring state \( s' \) is decided by velocity of the medium flow and thrust-producing action.
- \( v_{s,t}^r = \) Current velocity of the robot (or USV)
  - \( v_{s,t}^{r,o} = \) Component along orthogonal direction
  - \( v_{s,t}^{r,D} = \) Component along desired direction
- \( v_{s,t}^m = \) Current velocity of the medium
  - \( |v_{s,t}^m| \sin(\psi_e) = \) Component along orthogonal dir.
  - \( |v_{s,t}^m| \cos(\psi_e) = \) Component along desired dir.
- Forward Velocity
  \[
  |v_{s,t}^f| = |v_{s,t}^{r,D}| + |v_{s,t}^m| \cos(\psi_e)
  \]
- Assumption: Low level controllers of the vehicle are able to maintain the desired dir.
Admissible Heuristic #2

Projected distance towards goal

Selected Action

Free-flow Action

Actuation Action

Cost incurred per unit distance advancement towards the goal

\[
C^l = \frac{C^a_t}{|v_{r,s,t}| + |v_{m,s,t}^m| \cos(\psi_e)} > C^l = \frac{C^m_t}{|v^m_m| \cdot \cos(\psi_e)}
\]
Admissible Heuristic #3

- Heuristic #2 does not account for the orthogonal deviation from the desired path to reach the goal.
- Free-flow actions have to compensate for its deviation from the desired direction to reach the goal.

\[ d_c = \sqrt{d^2 + (|v^m| \cdot \sin(\psi_e) \cdot \delta t)^2} \]

\[ |v^f_c| = |v^r| + |v^m|_{max} \cdot \cos(\alpha) \]

where \( \alpha = \tan^{-1}\left[(|v^m|_{max} \cdot \sin(\psi_e) \cdot \delta t)/d\right] \)

\[ t_c = d_c/|v^f_c| \]

- New cost incurred per unit length advancement towards the goal for free-flowing action is given by:

\[ C^l = \frac{t_c \cdot C^t_a + C^t_m \cdot \delta t}{d + |v^m| \cdot \cos(\psi_e) \cdot \delta t} \]
Simulation Results

- Cost of actuation action $C_a^t = 6$ per min
- Cost of free-flowing action $C_m^t = 1.2$ per min
- Discrete time interval $\delta t = 10$ min
- Experimental Scenarios A:
  - Medium flow with constant magnitude of 6 m/s
    - Constant flow direction at $30^\circ$ and $170^\circ$
    - Rotating Medium flow at rate of $0.1^\circ$ per min with initial direction set at $330^\circ$
    - Rotating Medium flow at rate of $0.2^\circ$ per min with initial direction set at $310^\circ$

<table>
<thead>
<tr>
<th>Scenarios with Constant Magnitude of Flow</th>
<th># States Expanded</th>
<th>% Reduction in States Expanded</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># 1</td>
<td># 2</td>
</tr>
<tr>
<td>Constant Direction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30$^\circ$</td>
<td>196320</td>
<td>71321</td>
</tr>
<tr>
<td>170$^\circ$</td>
<td>459860</td>
<td>9041</td>
</tr>
<tr>
<td>Rotating Direction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rate 1$^\circ$/s</td>
<td>48864</td>
<td>15590</td>
</tr>
<tr>
<td>Rate 2$^\circ$/s</td>
<td>58893</td>
<td>14649</td>
</tr>
</tbody>
</table>
Simulation Results (cont.)

- **Experimental Scenarios B:**
  - Medium flow with randomly generate magnitude profile
    - Constant flow direction at $30^\circ$ and $90^\circ$
    - Randomly generate direction profile with maximum change of $5^\circ$ in each discrete time interval $\delta t = 10$ min
    - Rotating Medium flow at rate of $0.1^\circ$ per min with initial direction set at $330^\circ$
    - Rotating Medium flow at rate of $0.2^\circ$ per min with initial direction set at $310^\circ$

<table>
<thead>
<tr>
<th>Scenarios with Random Magnitude</th>
<th># Avg. States Expanded (in order of 1000s)</th>
<th>% Reduction in States Expanded # 2 with # 1</th>
<th># 3 with # 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># 1</td>
<td># 2</td>
<td># 3</td>
</tr>
<tr>
<td>Constant Direction $30^\circ$</td>
<td>186.62</td>
<td>56.59</td>
<td>3.83</td>
</tr>
<tr>
<td>Constant Direction $170^\circ$</td>
<td>363.71</td>
<td>17.73</td>
<td>7.14</td>
</tr>
<tr>
<td>Random Direction Init at $0^\circ$</td>
<td>323.08</td>
<td>53.75</td>
<td>32.37</td>
</tr>
<tr>
<td>Random Direction Init at $45^\circ$</td>
<td>270.45</td>
<td>48.63</td>
<td>37.23</td>
</tr>
<tr>
<td>Rotating Direction Rate $1^\circ/s$</td>
<td>336.75</td>
<td>52.01</td>
<td>24.13</td>
</tr>
<tr>
<td>Rotating Direction Rate $2^\circ/s$</td>
<td>215.75</td>
<td>28.45</td>
<td>13.32</td>
</tr>
</tbody>
</table>

- The developed admissible heuristic improves the computational performance of the planner by approximately 90% as compared to baseline heuristic
Weather forecast can be used to determine the optimal time when the vehicle should start the mission $t_{start}$.

Deliberative lattice-based A* planner compute optimal cost ($C^*$) from start to goal state at each sampled time $t_{start}$.

The optimizer adaptively samples the time and determines the resolution-optimal start time.
Simulation Results

**A.1**  
$t_{\text{start}} = 20 \text{ min}$  
$c^*(s_l, s_G) = 660$

**A.2**  
$t_{\text{start}} = 130 \text{ min}$  
$c^*(s_l, s_G) = 804$

**A**  
$t_{\text{start}} = 90 \text{ min}$  
$c^*(s_l, s_G) = 564$

**B**  
$t_{\text{start}} = 120 \text{ min}$  
$c^*(s_l, s_G) = 576$

**C**  
$t_{\text{start}} = 20 \text{ min}$  
$c^*(s_l, s_G) = 504$

**D**  
$t_{\text{start}} = 50 \text{ min}$  
$c^*(s_l, s_G) = 540$
Planning for Target Following with Motion Prediction

Presented at Conference IDETC 2012
Presented at Conference IROS 2013
Published in Journal Autonomous Robots 2014
Safe and efficient following of time varying motion goal $x_G = [x, y, \theta, v]^T$ in environment with obstacles presents multiple challenges

- USV following capability is influenced by its dynamics $\dot{x} = f_U(x, u)$, motion characteristics of motion goal, and distribution of obstacles
- USV does not know trajectory of target in advance
- USV needs to keep motion goal within specified distance range $(r_{min}, r_{max})$
- USV needs to maintain sufficient velocity and have capability to negotiate sharp turns
Approach

Dense action set
Sparse action set

Trajectory Planner: Multi-resolution

Control Action Set

Motion Goal Prediction
Experimental Results

https://youtu.be/LTwZwjT8o58
Reactive Trajectory Planner

Presented at PAMR Workshop held at Conference ICRA 2014
Published in Journal of Ocean Engineering 2016
Local, Adaptive Sampling-Based Trajectory Planner

- Developed generalized Velocity Obstacles based planner that computes a dynamically reachable, COLREGs-compliant motion goal
  - Generalizes state-of-the-art OA approaches [Wilkie, D. et al., 2009; Berg et al., 2012; Bareiss et al., 2013] to any non-linear, non-holonomic dynamic model and controller
  - Determines motion goals reachable along collision-free state space trajectories through resolution adaptive sampling
  - Can handle constraints on lower-level control input to ensure COLREGs
Local Trajectory Planner: Adaptive Sampling Based Search

- Expanded set of motion goals in classical, uniform sampling-based local planning
- Expanded set of motion goals in adaptive, non-uniform sampling-based local planning
- Adaptive sampling based on temporal-spatial distribution of obstacle vessels and history of detected collisions during the search
  - Control-imposed obstacle regions adaptively expanded

- Two types of sampling:
  1) Prioritized sampling of entire set of state space trajectories in each sampling cycle
  2) Sampling of individual state space trajectories for collision detection
Cost computation of sampled motion goals

\[ c(x_G) = \omega t_c + (1 - \omega)t_g + c_{\text{col,soft}} + c_{-\text{COLREGS}} \]

- \( t_c \) is the true time to reach the motion goal
- \( t_g \) is heuristic estimate of the time to reach goal state
- \( \omega \) controls the breadth of the sampling-based search
- \( c_{\text{col,soft}} \) is the cost for entering the soft obstacle region
Simulation Setup

- 1000 randomly generated evaluation scenarios
  - 200 x 200 m test scene
  - 3 DOF model of USV dynamics
  - USV’s max. speed is 2.5 m/s
  - 20 civilian, randomly distributed vessels with varying speed up to 3 m/s
  - following COLREGs was not considered

- Intel(R) Core(TM) i7-2600 CPU @ 3.4 GHz computer with 8GB RAM
Simulation Results (Cont.)

- Comparison of collision rate and search time of the **full sampling-based OA planner** to Velocity Obstacles [Fiorini and Shiller, 1998]

1) Planner considers vehicle’s dynamics and control in the search for a motion goal
2) However, its collision rate is higher than that of the Velocity Obstacles!
3) Planning is ~2.5 times slower than that of the Velocity Obstacles
4) Time to reach the goal is comparable to the time taken by Velocity Obstacles
Comparison of collision rate and search time of the full and adaptive sampling-based OA planners to Velocity Obstacles [Fiorini and Shiller, 1998]

1) Planners consider vehicle’s dynamics and control in the search for a motion goal
2) The search speed of the adaptive sampling-based planner comparable to the speed of the Velocity Obstacles
3) This results in more than 50% reduction in the collision rate
4) Time to reach the goal is comparable to the time taken by Velocity Obstacles
For more information please contact:
Satyandra K. Gupta
(guptask@usc.edu)