SMOOTHING VORONOI-BASED PATH WITH MINIMIZED LENGTH AND VISIBILITY USING COMPOSITE BEZIER CURVES

ILYA MAKAROV AND PAVEL POLYAKOV

NATIONAL RESEARCH UNIVERSITY HIGHER SCHOOL OF ECONOMICS

MOSCOW, RUSSIA
VORONOI-BASED NAVIGATION MESH

- Let $P = \{p_0, p_1, \ldots, p_n\}$ be a set of points called sites
- Let $VD(p_i) = \{x: |p_i - x| \leq |p_j - x|, \forall j \neq i, x \in \mathbb{R}^2\}$ be a polygon of a mesh
- Let a union of connected polygons be a **Voronoi surface**
- Then a union of Voronoi surfaces is a **Voronoi-based navigation mesh**
VORONOI-BASED NAVIGATION MESH

- Provides an opportunity to find paths considering tactical properties
- Allows to solve such problems as:
  - Predicting actions of opposing team
  - Searching for sniper and cover positions
  - Adopting to dynamically changing situation
- Helps to smooth produced paths
- Helps to track movement as a sequence of polygons
CONSTRUCTION PIPELINE

Voronoi diagram construction → Quad tree construction → Projection and obstacle detection → Voronoi links building

Voronoi diagram construction → Quad tree construction → Projection and obstacle detection → Static properties calculation → Ready

Voronoi diagram construction → Quad tree construction → Projection and obstacle detection
Voronoi diagram construction

Quad tree construction

Projection and obstacle detection

Voronoi links building

Static properties calculation

Ready

- Fortune’s Algorithm, $\Theta(n \log n)$
- Stored in DCEL, $\Theta(n)$ memory footprint

- Used for a solution of a point location problem or
- Finding polygons belonging to a specified area

- Projecting a diagram to geometry
- Performing collision checks along edges

CONSTRUCTION PIPELINE
CONSTRUCTION PIPELINE

- Optimizations:
  - One end of a link should be a border polygon
  - Candidates for the second end are found using quad trees

- Link candidates are eliminated if:
  - Height difference is too high
  - Segment of polygons’ sites intersects edge of the border polygon with is not near the border (in order to prevent redundant links)
TACTICAL PROPERTIES CALCULATION

- **Let visibility** be a value from 0 to 1 indicating an amount of area visible from a polygon within a given range.

- The sum of areas of visible polygons is divided by some predetermined constant and then clamped to [0, 1] range.

- Several line collision checks between a pair of polygons may be performed in order to distinguish a case of partial visibility.
TACTICAL PROPERTIES CALCULATION

• **Visibility measure allows us to:**
  • **Find covers and predict where opponents could hide**
  • **Search for paths moving along which will be detected with the lowest probability according to a map topology**

• **Other tactical properties consist of:**
  • **Influence map**
  • **Frag map**
  • **Danger map**
  • **Loot map**
  • **Sniper positions**
PATH PLANNING PIPELINE

Querier

Find path from A to B

Voronoi-based navigation mesh

Request querier’s preferences

Request penalties similar to Markov’s chain process

A*

Building a Bezier curve

Return a sequence of points

Post processing stage

\[ d_j = \text{distance}(v_j, \text{line}(p_{i-1}, p_{i+1})), w_j = \frac{\text{rand}(1, b)}{d_j}, j = 1, 2 \]

\[ p_i = \frac{v_1 \cdot w_1 + v_2 \cdot w_2}{w_1 + w_2} \]
BUILDING A COMPOSITE BEZIER CURVE

A sequence of points → Insert additional points → Split into pieces with collision-free convex hull

- Strategy:
  \[
  \sum_i \text{distance(Piece}[i].\text{FirstPoint, Piece}[i].\text{LastPoint}) \rightarrow \min
  \]

- Complexity: \(\theta(n^3)\)

- Ray casts: \(\theta(n^2)\)

- Removal of crowded points
- Ensure C1 continuity
- Ready
EXPERIMENT AND CONCLUSION

• **Comparison with the shortest path length**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD, %</td>
<td>5.9</td>
<td>12.5</td>
<td>27</td>
<td>37.5</td>
</tr>
<tr>
<td>VD, %</td>
<td>0.3</td>
<td>1</td>
<td>0.7</td>
<td>2.1</td>
</tr>
</tbody>
</table>

• [1, 2] **Piecewise path with visibility penalty multiplier equaled 0 and 10;**

• [3, 4] **Smoothed path with visibility penalty multiplier equaled 0 and 10.**
THANKS FOR ATTENTION

Ilya Makarov: iamakarov@hse.ru
Pavel Polyakov: polyakovpavel96@gmail.com