Edge N-Level Sparse Visibility Graphs:
Fast Optimal Any-Angle Pathfinding Using Hierarchical Taut Paths

Shunhao Oh
ohoh@u.nus.edu

Hon Wai Leong
leonghw@comp.nus.edu.sg

Department of Computer Science
National University of Singapore
Any-Angle Pathfinding

Problem Definition
Any-Angle Pathfinding

Problem Definition
Any-Angle Pathfinding

Problem Definition

Grid
Any-Angle Pathfinding

Problem Definition

Blocked Tile
Any-Angle Pathfinding
Problem Definition
Any-Angle Pathfinding

Problem Definition
Any-Angle Pathfinding

Problem Definition
Any-Angle Pathfinding

Problem Definition
8-Directional Path

Total Length: $4 + 3\sqrt{2} \approx 8.243$
Any-Angle Path

Total Length: $2 + \sqrt{13} + \sqrt{5} \approx 7.842$
Visibility Graphs
Visibility Graph
Visibility Graph
Visibility Graph

Visibility Graphs
Visibility Graph

Visibility Graphs
Visibility Graph

Visibility Graphs
Taut Paths
Taut Paths
Optimal Paths must be Taut!
Non-Taut Path
Taut Path
Taut Path

Heading change

Taut Paths
Tautness Checks
Tautness Checks

We need only check one tile.
Tautness Checks

We need only check one tile.
Tautness Checks

We need only check one tile.

Cannot be taut!
Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Visibility Graph
Sparse Visibility Graph
Sparse Visibility Graph

Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Sparse Visibility Graphs
Visibility Graph

Sparse Visibility Graphs
Sparse Visibility Graph
Sparse Visibility Graph
Sparse Visibility Graph
Sparse Visibility Graph
Sparse Visibility Graphs
Sparse Visibility Graph

**Theorem 1**

Every optimal path can be represented using only edges in the sparse visibility graph.
Sparse Visibility Graph

**Theorem 2**

Every edge in the sparse visibility graph is part of some optimal path between some pair of points.
(where neither point is an endpoint of the edge)
Sparse Visibility Graph

Theorem 2

Every edge in the sparse visibility graph is part of some optimal path between some pair of points. (where neither point is an endpoint of the edge)

i.e. every edge is “essential”
Sparse Visibility Graph
Sparse Visibility Graph
Sparse Visibility Graphs
Sparse Visibility Graph
Finding Visible Neighbors with Line-of-Sight Scans
Line-of-Sight Scans
Neighbors?
Line-of-Sight Scans
Line-of-Sight Checks

Line-of-Sight Scans
Line-of-Sight Scans
Interval Search
Line-of-Sight Scans
Line-of-Sight Scans
Line-of-Sight Scans
Line-of-Sight Scans
Line-of-Sight Scans
Line-of-Sight Scans
All-Direction Line-of-Sight Scans

VS

Taut-Direction Line-of-Sight Scans
All-Direction Line-of-Sight Scans
All-Direction Line-of-Sight Scans
All-Direction Line-of-Sight Scans
All-Direction Line-of-Sight Scans
Taut-Direction
Line-of-Sight Scans
Taut-Direction Line-of-Sight Scans
Taut-Direction Line-of-Sight Scans

Line-of-Sight Scans
Taut-Direction Line-of-Sight Scans
Taut-Direction Line-of-Sight Scans
All-Direction Line-of-Sight Scan

Line-of-Sight Scans
Line-of-Sight Scans
Sparse Visibility Graph Construction: Taut-Direction Line-of-Sight Scans
Inserting Start & Goal Points: All-Direction Line-of-Sight Scans
Inserting Start & Goal Points: All-Direction Line-of-Sight Scans
Comparison of Line-of-Sight Algorithms
Comparison of Line-of-Sight Algorithms
(Using Construction Time)
Comparison of Line-of-Sight Algorithms

Visibility Graph

VG_C Line-of-Sight Checks (All-Pairs)

VG_RPS Rotational Plane Sweep Algorithm

VG_S Line-of-Sight Scans
Comparison of Line-of-Sight Algorithms

Visibility Graph

VG_C   Line-of-Sight Checks (All Pairs)

VG_RPS Rotational Plane Sweep Algorithm

VG_S   Line-of-Sight Scans

Sparse Visibility Graph

SVG    Sparse Visibility Graph (Line-of-Sight Scans)
Benchmark Maps

Game Maps

Random Maps
Construction Time

Benchmarks (Game) Construction Time
Construction Time (ms)

Remove VG_RPS
Construction Time (log10, ms)
Construction Time (log10, ms)

![Bar chart showing log scale (log10) for construction time across different variables and datasets.](chart.png)
Construction Time

Benchmarks (Random) Construction Time
Construction Time (ms)

Remove VG_RPS
Construction Time (log10, ms)

log scale (log10)
Construction Time (log10, ms)
Edge N-Level Sparse Visibility Graphs
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Idea:
Some edges are only used as a “Second hop from the start” or a “Second hop from the goal”
Thus:

Prune away all edges that do not lead to another edge via a taut path.
Thus:

Prune away all edges that do not lead to another edge via a taut path

Mark them as "Level 1"
Sparse Visibility Graph

Edge N-Level Sparse Visibility Graphs
Sparse VG Algorithm

Edge N-Level Sparse Visibility Graphs
Then:

Do the same thing again. These edges are at most the third hop from the start or goal.
Then:

Do the same thing again. These edges are at most the third hop from the start or goal.

Mark them as “Level 2”
Level 1 edges removed
Level 2 edges removed
Level 3 edges removed
Level 4 edges removed
Edge N-Level SVG

Edge N-Level Sparse Visibility Graphs
Theorem

Assuming all edges have finite level,
Theorem

Assuming all edges have finite level,

Any taut path between the start and goal will be of the form:
Theorem

Assuming all edges have finite level,

Any taut path between the start and goal will be of the form:

\[ e_1 e_2 \ldots e_k e'_{k+1} \ldots e'_{n-1} e'_{n} \]
Theorem

Assuming all edges have finite level,

Any taut path between the start and goal will be of the form:

\[ e_1 e_2 \cdots e_k e'_{k+1} \cdots e'_{n-1} e'_n \]

where \( e_1 e_2 \cdots e_k \) have strictly increasing levels, and \( e'_{k+1} \cdots e'_n \) have strictly decreasing levels.
Edge N-Level Visibility Graph

Edge N-Level Sparse Visibility Graphs
Edge N-Level Visibility Graph

Edge N-Level Sparse Visibility Graphs
Edge N-Level Visibility Graph

Edge N-Level Sparse Visibility Graphs
Infinite-level edges?

Not all edges can be pruned this way. The remaining edges are called Level-W edges.
Level-W Edges

Edge N-Level Sparse Visibility Graphs
Note:

An edge is Level-W if and only if it is part of some taut cycle.
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Theorem

Any taut path between the start and goal will be of the form:
Theorem

Any taut path between the start and goal will be of the form:

\[ e_1 e_2 \cdots e_{k_1} w_{k_1+1} \cdots w_{k_2} e'_{k_2+1} \cdots e'_{n-1} e'_n \]
Theorem

Any taut path between the start and goal will be of the form:

\[ e_1 e_2 \cdots e_{k_1} w_{k_1+1} \cdots w_{k_2} e'_{k_2+1} \cdots e'_{n-1} e'_n \]

where \( e_1 e_2 \cdots e_k \) have strictly increasing levels, and \( e'_{k_2+1} \cdots e'_n \) have strictly decreasing levels and \( w_{k_1+1} \cdots w_{k_2} \) are all level-W.
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Skip Edges
Level-W Edges
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
"Degree 2" vertices
Skip Edge

Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Note: The cost of the edge is not the length of this line.
Edge N-Level Sparse Visibility Graphs
“Skip Vertices”
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Algorithm
Three Phases

Step 1: Insertion
Step 2: Marking
Step 3: Search
Step 1: Insertion
Connect Start and Goal
Step 1: Insertion
Connect Start and Goal
(using Line-of-Sight Scans)
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Step 2: Marking

Mark all Edges Reachable by an Increasing Sequence or Level-W Edges.

Stop when you reach any Skip Vertex.

e.g. 1 3 4 5 8 W W W W W
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Mark from both the Start and the Goal
Step 3: Search

Search Only
Marked Edges & Skip Edges

Note: Use Taut A* Search
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Edge N-Level Sparse Visibility Graphs
Search Tree Comparison
Visibility Graph
Sparse Visibility Graph
Visibility Graph
Sparse Visibility Graph
Visibility Graph
Sparse Visibility Graph
Experiments
Running Time Comparison
Algorithms Compared

VG_C
VG_S
SVG
ENLSVG
Theta*
Anya16
SG16
Algorithms Compared

VG_C
VG_S
SVG
ENLSVG
Theta*
Anya16
SG16

Visibility Graphs Line-of-Sight Checks
Algorithms Compared

- **VG_C**
- **VG_S**
- **SVG**
- **ENLSVG**
- **Theta**
- **Anya16**
- **SG16**

Visibility Graphs
Line-of-Sight Scans
Algorithms Compared

VG_C
VG_S
SVG
ENLSVG
Theta*
Anya16
SG16

Sparse Visibility Graphs
(Line-of-Sight Scans)
Algorithms Compared

VG_C
VG_S
SVG
ENLSVG
Theta*
Anya16
SG16

ENLSVGs
(Line-of-Sight Scans)
Algorithms Compared

Theta* Algorithm
Nash et al. 2007

VG_C
VG_S
SVG
ENLSVG
Theta*
Anya16
SG16
Algorithms Compared

Anya (2016 Version)

Harabor et al. 2016

VG_C
VG_S
SVG
ENLSVG
Theta*
Anya16
SG16
Algorithms Compared

Speeding Up A* Search on Visibility Graphs Defined over Quadtrees

Shah & Gupta, 2016
Running Time Comparison
Running Time Comparison

All running times are in milliseconds
Running Time

Benchmarks (Game)
Running Time

Benchmarks (Game)
Running Time

Benchmarks (Game)
Running Time
Benchmarks (Game)

Remove SG16
Remove Theta*
Running Time

Benchmarks (Game)

Remove SG16
Remove Theta*
Remove VG_C
Running Time

Benchmarks (Random)
Running Time

Benchmarks (Random)

random10, 512x512

random40, 512x512
Running Time

Benchmarks (Random)

Note: SG16 too slow on random maps
Cave Maps
Generated using Cellular Automata
(Johnson, Yannakakis, and Togelius 2010)
Running Time

Cave Maps

30% Blocked
4000x4000

45% Blocked
4000x4000
Running Time
Cave Maps

30% Blocked
Running Time
Cave Maps – 30% Blocked
Running Time

Cave Maps – 30% Blocked

Remove SG16
Running Time

Cave Maps – 30% Blocked

Remove SG16
Remove Theta*
Running Time

Cave Maps – 30% Blocked

Remove SG16
Remove Theta*
Remove VG_C
Running Time

Cave Maps

45% Blocked
Running Time

Cave Maps – 45% Blocked
Running Time
Cave Maps – 45% Blocked

Remove SG16
Remove Theta*
Remove VG_C
Running Time

Upscaled Benchmarks
Running Time

Upscaled Benchmarks

Upscaled Map (wc3, 512x512)

Map Size: 4096x4096

Cellular Automata used for smoothing
Running Time
Upscaled Benchmarks
Running Time
Upscaled Benchmarks

Remove SG16
Remove Theta*
Remove VG_C
Tiled Benchmarks
Running Time

Tiled Benchmarks

8x8 Tiled Map
(wc3, 512x512)

Map Size:
4096x4096
Running Time

Tiled Benchmarks

![Graph showing running time for different tiled benchmarks and input sizes.]

- SG16
- Theta*
- Anya16
- VG_C
- VG_S
- SVG
- ENLSVG

Input Sizes:
- 2048x2048
- 4096x4096
- 6144x6144
Running Time

Tiled Benchmarks

- Remove SG16
- Remove Theta*
- Remove VG_C
Running Time Breakdown (ENLSVG Algorithm)
Running Time Breakdown (ENLSVG Algorithm)

Three Main Components

Insertion

Marking

Search
Running Time Breakdown
(ENLSVG Algorithm)
Running Time Breakdown
(ENLSVG Algorithm)

Insertion
Marking
Search
Running Time Breakdown (ENLSVG Algorithm)

- Insertion
- Marking
- Search

Line-of-Sight Scan
Running Time Breakdown
(ENLSVG Algorithm)

- Insertion
  - Line-of-Sight Scan
- Marking
  - BFS/DFS
- Search
Running Time Breakdown
(ENLSVG Algorithm)

- Insertion
  - Line-of-Sight Scan
- Marking
  - BFS/DFS
- Search
  - A* Search
Running Time Breakdown (ENLSVG Algorithm)
Running Time Breakdown (ENLSVG Algorithm)

Scaled Maps: Mostly Insertion/Marking
Running Time Breakdown
(ENLSVG Algorithm)

Tiled Maps: Mostly Search
Running Time

Maze Maps
Running Time
Maze Maps

2000x2000 Maze
Corridor width 2

Spanning Tree
\( n - 1 \) edges
Running Time

Maze Maps: Spanning Tree

Note:
SG16 too slow on mazes
VG_C not tested
Running Time

Maze Maps: Spanning Tree

Remove Theta*
Remove Anya16
Running Time

Maze Maps

2000x2000 Maze Corridor width 2

Approximately 0.0001n + n - 1 edges
Running Time

Maze Maps: $+0.0001n$ edges

Note:
SG16 too slow on mazes
VG_C not tested
Running Time

Maze Maps: $+0.0001n$ edges
Running Time

Maze Maps: $+0.0001n$ edges

Remove Theta*
Remove Anya16
2000x2000 Maze Map
Corridor width 2

Approximately
0.001n + n - 1
edges
Running Time

Maze Maps: $+0.001n$ edges

Note:
SG16 too slow on mazes
VG_C not tested
Running Time

Maze Maps: \( +0.001n \) edges

Remove Theta*
Running Time

Maze Maps: $+0.001n$ edges

Remove Theta*
Remove Anya16
Running Time

Maze Maps

ENLSVG
Search Tree

SVG
Search Tree
Implementation

github.com/Ohohcakester/Any-Angle-Pathfinding
Implementation

github.com/Ohohcakester/Any-Angle-Pathfinding

Google for "Any Angle Pathfinding"
Edge N-Level Sparse Visibility Graphs: Fast Optimal Any-Angle Pathfinding Using Hierarchical Taut Paths

Shunhao Oh
ohoh@u.nus.edu

Hon Wai Leong
leonghw@comp.nus.edu.sg

Department of Computer Science
National University of Singapore

Implementation:
github.com/Ohohcakester/Any-Angle-Pathfinding
extra
Benchmarks (Game)
Construction Time
Construction Time (ms)
Remove VG_RPS
Construction Time (log10, ms)

log scale (log10)
Benchmarks (Random)
Construction Time
Construction Time (ms)
Construction Time (ms)

Remove VG_RPS
Construction Time (log10, ms)

log scale (log10)
Average Vertex Degree on Random Maps
Sparse Visibility Graphs

Silhouette Graphs

N-Level Subgoal Graphs

Contraction Hierarchies

Anya

Hierarchical Taut Paths

Line-of-Sight Scans

ENLSVGS
Roadmap

Silhouette Graphs

Level $\geq 0$

ENLSVG

Tangent Graphs

Level $W$
ACKNOWLEDGEMENT

This presentation benefitted from

**PowerPointLabs**

a PowerPoint plugin for creating

**better presentations with less effort.**

PowerPointLabs

is available for free at

http://PowerPointLabs.info