Computing Exact Arbitrary Clearance for Navigation Meshes

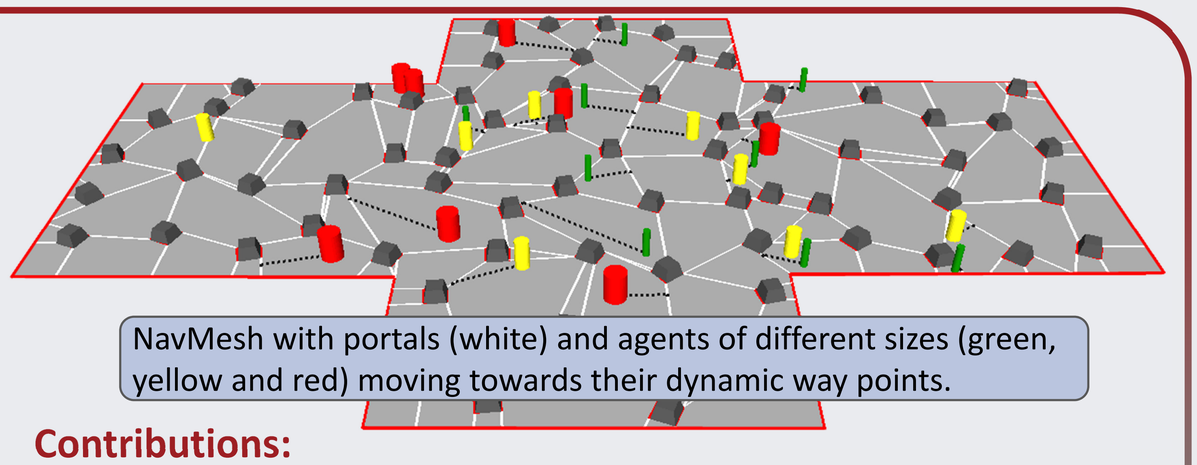
R. Oliva, A. Beacco & N. Pelechano
Universitat Politècnica de Catalunya

Character navigation in virtual environments is mostly handled by a combination of path planning with local movement algorithms. Clearance should be taken into consideration when choosing a path, but also when deciding the location of way points within portals. Previous work has considered clearance for path planning, but it is ignored when assigning way points within portals. Paths with clearance, do not necessarily guarantee that characters can walk through collision-free way points in portals. In this work we present a method for calculating ***clearance in navigation meshes*** *(NavMeshes) consisting of convex cells of any type, as well as a* ***novel method to calculate portals with clearance****. We also introduce a new method to* ***dynamically locate attractors*** *over portals based on current trajectory, destination, and clearance.*

Introduction

- Clearance in NavMeshes is taken into account by most path planning algorithms, to guarantee that the character can walk between way points without colliding with the static environment.
- Previous work is either bounded to a specific amount of clearance, or only work with a specific type of NavMesh (e.g: triangular meshes, medial axis).
- Our work aims at dealing with an arbitrary amount of clearance, and also at assigning collision-free way points spread over the whole length of portals.
- We have integrated both our local movement algorithm and exact arbitrary clearance computation into *NEOGEN* [1].

[1] R. Oliva, N. Pelechano. NEOGEN: Near optimal generator of navigation meshes for 3d multi-layered environments. *Computer & Graphics*. vol. 37, no.5, pp 403-412. Elsevier. August 2013.



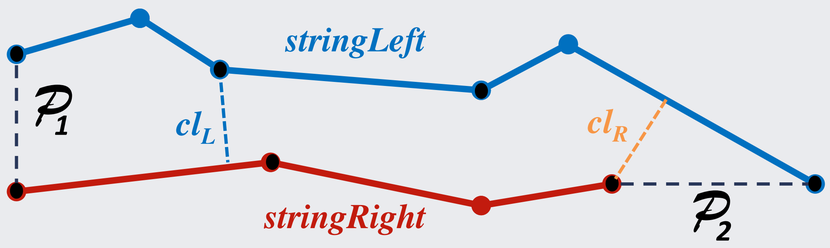
NavMesh with portals (white) and agents of different sizes (green, yellow and red) moving towards their dynamic way points.

Contributions:

1. A novel method for calculating clearance in general NavMeshes.
2. A novel method to calculate portals with clearance.
3. A new method that assigns attractors to portals dynamically using the whole length of the portal with clearance to avoid forcing all agents to walk through a fixed location.

Clearance Value of a Cell

Given a cell $\mathcal{C}$, an entry portal $\mathcal{P}_1$ and an exit portal $\mathcal{P}_2$, we classify the obstacle edges of the cell into edges to the left (*leftString*) and edges to the right (*rightString*) (respect to the crossing path from entry to exit portal). Cells do not need to be strictly convex [1].



Clearance calculation due to small allowed concavities (cl_L) or by an endpoint of a portal (cl_R).

The algorithm iterates over every notch (concave vertex) in ***leftString*** looking for the closest edge in ***rightString*** and calculating the clearance for the notch. The endpoints of each string are also treated as notches. The clearance value of the left string cl_L is the minimum of the notches' clearance. Likewise, we calculate the clearance value of the right string cl_R. And finally, the clearance value of the described path is computed as follows:

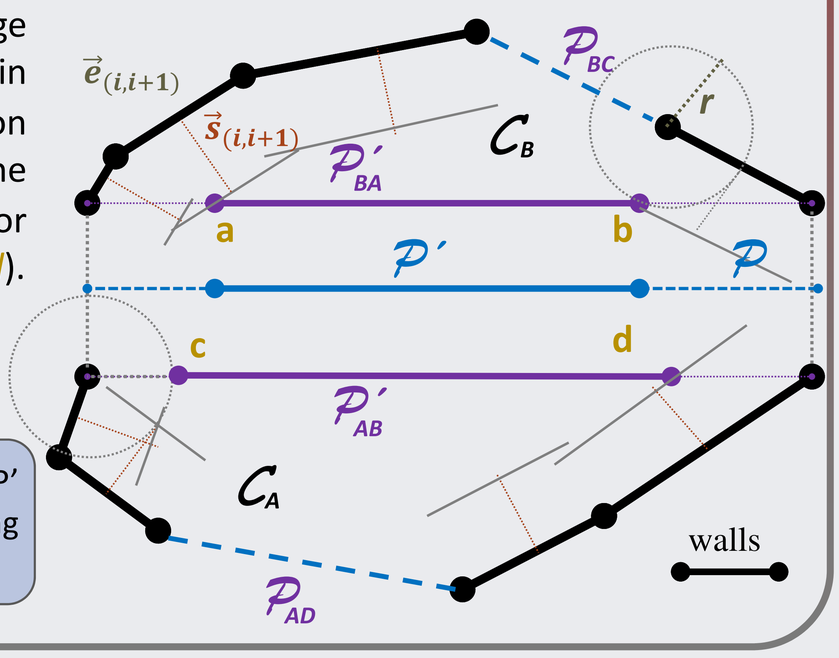
$$cl(\mathcal{P}_1,\mathcal{P}_2) = min(cl_L, cl_R)$$

Finding portals $\mathcal{P}'$ with clearance

Split $\mathcal{P}$ in two collinear segments $\mathcal{P}_{AB}$ and $\mathcal{P}_B$ and treat them separately to iteratively shrink them to finally calculate: $\mathcal{P}' = \mathcal{P}'_{BA} \cap \mathcal{P}'_{BA}$

The iterative shrinking process checks for:

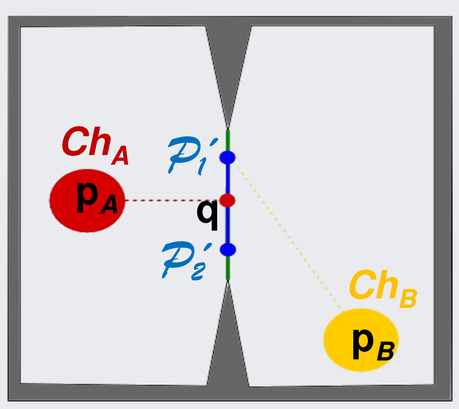
1. Limitation by *r*: shrink each endpoint of $\mathcal{P}_{AB}$ by *r* (see *c*).
2. Limitations by other portals: An endpoint of $\mathcal{P}_{BC}$ is at a distance ≤ *r* from $\mathcal{P}_{BA}$. Check if a circumference centered in the endpoint of $\mathcal{P}_{BC}$ intersects with $\mathcal{P}'_{BA}$ (see *b*).
3. Limitation by edges of cell: displace each edge $\vec{e}_{(i,i+1)}$ a distance *r* in the shrinking direction $\vec{s}_{(i,i+1)}$ (interior of the cell) and check for intersection (see *a* & *d*).



Shrinking portal P into P' with a,b,c and d showing the different cases.

Computing Dynamic Way Points

As characters approach portals, way points are assigned dynamically either as their orthogonal projection over the portal or as the farthest away end point of the portal.
Local movement will steer characters towards different way points .



Examples of two dynamically generated way points over the portal with clearance resulting from shrinking the original portal. Ch_A has **q** as next way point, while Ch_{AB} has $\mathcal{P}'_1$

Results & Conclusions

- ✓ We have presented general technique to compute paths with arbitrary clearance, as well as a method to pre-calculate portals with clearance and dynamic way points (average time = 4.5µs.) which provides natural looking trajectories.
- ✓ Tested 1000 path queries with A* in different scenarios and character sizes. Since clearance helps to prune the search, on average the results were:

clearance	0.5	1.0	1.5
times faster	1.06	1.15	2.27

(Intel Core 2 Quad Q9300 @ 2.50GHz, 4GB of RAM).

ViRVIG Visualització, Realitat Virtual i interacció Gràfica
Anaheim 19th – 21st July SCA 2013 Symposium on Computer Animation the future is animated