Killzone's AI: Dynamic Procedural Tactics

Arjen Beij & William van der Sterren
Killzone’s AI: Dynamic Procedural Tactics

Arjen Beij
Guerrilla Games
Amsterdam, the Netherlands
Arjen.Beij@guerrilla-games.com

William van der Sterren
CGF-AI
Veldhoven, the Netherlands
william@cgf-ai.com
Contents

- Dynamic procedural tactics
- Position picking
- Position Evaluation functions
- Tactical path-finding
- Suppression Fire
- World Representation
- Conclusions
Dynamic procedural tactics =

- Having the rules and concepts to dynamically compute solutions to tactical problems
Common approach

- Level designer placed hints
- Triggered scripted behavior
Dynamic Procedural Tactics

Game-play benefits
- Works for any number of players
- Fights anywhere

Production benefits
- Reuse of AI behavior
- Quick level prototyping
- Roll out improvements game wide
Dynamic procedural tactics

Position picking

Position Evaluation functions

Tactical path-finding

Suppression Fire

World Representation

Conclusions
Position Picking

- Picking a good position is half the battle in ‘fire & maneuver’ combat

- Account for dozens of (dynamic) factors

- How to pick that position?

Keywords:
- Outside danger zone
- Wall Hugging
- Partial Cover
- Nearby Cover
- Line of Fire
- Proximity
- Preferred Range

March 10, 2005
Game Developer Conference

Killzone’s AI: Dynamic Procedural Tactics

Guerrilla Games - 8
Position Evaluation Functions

Determining the most appropriate position
- Given $n$ potential destinations
- To compare positions, need a single value

Use position evaluation function
- Weighted sum
- Combines basic factors
Example: Picking an attack position

- Multiple threats
  - Attack primary threat
  - Cover from other threats preferred
- Partial cover from primary
- Proximity
- Preferred fighting range

Combine into a single value per position
Example: Picking an Attack Position 2

Initial situation: three threats, some walls and a waypoint graph
Example: Picking an Attack Position 3

Selected nearby waypoints, annotated with proximity (weight 20)
Example: Picking an Attack Position 4

Annotations for LoF to primary threat (40 if partial cover, 20 otherwise)
Example: Picking an Attack Position 5

Annotations for cover from secondary threats (weight 20)
Example: Picking an Attack Position 6

Annotations positions inside preferred fighting range (weight 10)
Example: Picking an Attack Position 7

Adding up all the annotations yields the most promising attack position.
Example: Picking an Attack Position

Single threat  Multiple threats

[Images of different attack positions for single and multiple threats]
Killzone's AI: Dynamic Procedural Tactics

Guerrilla Games - 18

Configurable and Dynamic

- Configurable
  - Action: What kind of position?
  - Personality: What factors are important?

- Dynamic
  - Current situation: Where are the threats?
  - Squad: Additional constraints

Position evaluation function
Contents

- Dynamic procedural tactics
- Position picking
- Position Evaluation functions
- Tactical path-finding
- Suppression Fire
- World Representation
- Conclusions
Tactical Path-finding

Shortest paths often aren’t responsive or tactical

For example:
1. Paths traversing friendly lines-of-fire
2. Paths ignoring cover from hostile fire
3. Paths leaving the AI exposed unnecessarily
Position evaluations functions on the move...

- In A*, add to the costs of traveling a link:
  - Traversing a friendly lines-of-fire
  - Being under fire from one or more threats
Example: Tactical Path-finding 1

Costs to travel each waypoint link, and the resulting shortest path.
Example: Tactical Path-finding 2

Link costs now reflect being under fire, and the corresponding tactical path.
Example: Tactical Path-finding 3

shortest path versus tactical path
Suppression fire:

- Fire near hidden threats to pin them down

Not:

- Firing into the wall behind which the threat is hiding
- Firing into locations the threat cannot reach
Example: Suppression Fire 1

What we want

- Deny the threat use of good attack positions
- By pumping bullets into these positions

How we implement it using position evaluation

- Evaluate threat’s attack positions from his perspective(!)
- Select those attack positions we can fire into
Example: Suppression Fire 2

A hidden threat (left), attacker (right) and the intended suppression fire.
Example: Suppression Fire 3

Selecting the waypoints in close proximity to the presumed threat position
Example: Suppression Fire 4

Annotations (20, 30) for offering the threat a LoF to the attacker (right).
Example: Suppression Fire 5

Annotation of 20 for positions offering the threat nearby cover from the attacker.
Example: Suppression Fire 6

Selecting those positions with score $\geq$ 40 yields the suppress targets.
Example: Suppression Fire 7

Merging suppression targets that overlap in yaw and pitch.
Example: Suppression Fire 8

The attacker’s view of suppression targets near the hidden threat.
Contents

- Dynamic procedural tactics
- Position picking
- Position Evaluation functions
- Tactical path-finding
- Suppression Fire
- World Representation
- Conclusions
World Representation

Pre-conditions for dynamic procedural tactics

1. Navigation info, with fine enough resolution to represent cover locations
   - Waypoints, nav meshes, cells: any will do

2. Fast Line-of-Sight/Line-of-fire checks
Tactical decisions involve 100s of LoF checks
- Ray casts typically expensive
- So, offload most checks to something cheaper

Use a small pre-computed look-up table
- Multiple stances, minor movement
- Pessimistic about cover
- Polar representation

Killzone’s LoF table for 4000 waypoints: 64KB
Compression:
For every waypoint, per radial sector, record the largest distance from where an attacker within that sector can fire at the waypoint.

Intended Side-effect:
Those cover positions from where it would be easy to get into an attack position are often represented as having a line-of-fire.
Dynamic procedural tactics

Position picking

Position Evaluation functions

Tactical path-finding

Suppression Fire

World Representation

Conclusions
Conclusions

Killzone was released for PS2 late 2004

- Tactical position picking
- Tactical path-finding
- Suppression by infantry and mounted MGs
- Indirect fire for hand grenades and tank shells
- Single and multiplayer AI

- Up to 14 AI characters fighting simultaneously
Conclusions - Killzone’s AI

Pros
- Share/reuse behavior across games
- Quick level prototyping
- Can be combined with scripting
- Fights anywhere, anytime, SP or MP

Cons
- Harder to control in detail
- Harder to test
Conclusions - Game AI

Position evaluation functions are powerful
- Robust decisions with many (dynamic) inputs
- Different behavior through configuration
- Many tactics can be implemented through position evaluation

Dynamic procedural tactics
- Responsive AI behavior
- Within reach of many games
Further Info

Off-line

- Our proceedings paper provides more details, notably on the terrain representation and grenade handling


- Paul Tozour, *Using a Spatial Database for Runtime Spatial Analysis* in AI Game Programming Wisdom 2, Charles River Media, 2003


On-line

- William’s collection of links to on-line papers at: www.cgf-ai.com/links.html
Questions?