Terrain Reasoning for 3D Action Games

William van der Sterren

william@cgf-ai.com

IGDA Amsterdam chapter meeting, May 31, 2001
slides available soon at www.cgfai.com
Terrain Reasoning for 3D Action Games

William van der Sterren
william@cgf-ai.com

slides available at www.cgf-ai.com

Hi-lights from the GDC talk

some GDC do’s & don’ts
Terrain Reasoning

the AI capability
to take into account terrain
in its planning, decisions, actions, and communication

terrain = indoor + outdoor
Aim

why
■ terrain reasoning AI in 3D action games

how to
■ translate terrain concepts to algorithms
■ use terrain reasoning for real problems
■ add it to your game (AI)

briefly: when (not)
■ to use it (ideas, techniques, data structures)
William van der Sterren

researching and developing (tactical) AI
- CGF-AI, Veldhoven, the Netherlands
  - www.cgf-ai.com
- see next slides

senior scientist / architect + team leader
- Philips Electronics Research Labs
- embedded systems software

defense simulations
- naval simulations, TNO-FEL, 1995
Game Development Background

sQuad bot for Navy SEALs for QuakeI

96-97

Team tactical shooter AI (mod based)
Game Development Background

CGF 'bots' for Action Quake II

Team tactical shooter AI (mod based)
Game Development Background

GAMEdDevelopers Conference 2001

00

01

tactical AI technology
Contents

- intro
- case: track and predict opponent
- waypoints, and their use
- case: tactical terrain assessment
- pre-game and in-game analysis
- case: team maneuvers
- issues in implementing terrain reasoning
- conclusions

- GDC do’s and don’ts

Tailored to tonight’s audience:
# programmers?
# level designers?
# artists, producers, ... ?
# ‘mod’ developers?
Why Terrain Reasoning

3D level design
theater to shape interaction
every spot has a (tactical) purpose
stronghold, cover, avenues,…

10,000+ polygons

AI running around in shortest paths
Contents

- intro
- case: track and predict opponent
- waypoints, and their use
- case: tactical terrain assessment
- pre-game and in-game analysis
- case: team maneuvers
- issues in implementing terrain reasoning
- conclusions

- GDC do’s and don’ts
Case: Track and Predict Threat

This case illustrates:

- The need to represent terrain
- The use of waypoints
  - More than navigation aids
- Some smart, light-weight solutions
  - Prediction, corrected by path info
  - Prediction, corrected by line-of-sight info
  - Aiming for ‘reappear spots’ and portals
- Expressed in waypoints!
Case: Track and Predict Threat 2
Case: Track and Predict Threat

Problem
- predict position of opponent
- that moves out-of-sight
- scan and aim smart
- suppression fire? try a rocket?

Relevance
- one-on-one deathmatch
- friendly team member AI
- hard to shake off opponents
Case: Track and Predict Threat

Situation (top view)
Case: Track and Predict Threat

Where to look for the threat?
What to aim for?
Case: Track and Predict Threat 6

Option: assume threat static after moving out of sight (often incorrect)
Case: Track and Predict Threat

Option:
assume threat continues
last observed movement
(incorrect after a little while)
Case: Track and Predict Threat

If the AI
- assumes threat is static; or
- assumes threat continues movement

then the AI ignores information about
- (in)accessible terrain
- portals
- shortest paths
- (valid) lines of sight
- game play data

Often already available in 3D action games
Case: Track and Predict Threat

Waypoints\(^1\) describe
- accessible terrain + paths
- valid movement (speeds)
- properties of their local environment
  - door, window, water, tunnel, traffic, ...

Interwaypoint visibility approximates
- valid lines of sight

\(^1\): or similar constructs such as navigation cells, navigation meshes
Case: Track and Predict Threat

The waypoint grid (overlay)
Case: Track and Predict Threat

The waypoint grid (AI "view")
Case: Track and Predict Threat

Extrapolate movement along waypoints (valid paths, movement speed known).

Threat no longer “leaves” the house!
Case: Track and Predict Threat 12b

Same terrain, but using Half-Life / Unreal style grid:

Seems just as good, but...
Case: Track and Predict Threat

Use known line-of-sight for "negative" observations.
Exclude some likely threat movements.
Case: Track and Predict Threat

The AI can use the waypoint graph to

- restrict guesses to valid movements
- estimate the threats’ movement speed
- exclude visible potential movements that were not observed

Predict a likely path, for example

- path best aligned with last observed direction
- using per waypoint traffic statistics
- towards most attractive nearby objective
- towards strongest defensive position
Case: Track and Predict Threat

Where to aim?
- there where the threat likely reappears!

Good guesses
- visible spot closest to threat
- first visible spot on shortest path from threat to observer
- first visible spot on predicted path
- any nearby portal (door, window, hole, ...)
- easily computed using waypoint graph
Case: Track and Predict Threat

Threat turns to attack (1) or moves to nearest attack position (2), or continues movement (3)
Waypoints and their use

Waypoints

- more than a navigation aid
- mark terrain accessible to AI (and player)
- describe and represent surroundings
- express paths and travel time
- already in use by many AI’s
- in the order of 300 - 3000

- denser waypoint grid
  ➔ represents terrain better
Waypoints and their use 2

To support tracking and prediction

- per waypoint
  - outgoing neighbor waypoints
  - travel time to nearby waypoints
  - waypoints fully visible from here
  - (portal annotation)
  - (game play data)

and

- flood fill waypoint expansion; or
  - (short distance) path finding
Contents

- intro
- case: track and predict opponent
- waypoints, and their use
- case: tactical terrain assessment
- pre-game and in-game analysis
- case: team maneuvers
- issues in implementing terrain reasoning
- conclusions

GDC do’s and don’ts
Case: Tactical Assessment

this case illustrates

- Al tactical understanding of terrain
- use of captured game play data to
  - vary AI behavior
  - enhance Al tactical understanding of terrain (wrt to mission, objectives, spawn spots ...)
- expressing tactical concepts as evaluation functions per location
- incorporating analysis in the game
- expressed using waypoint graph
Case: Tactical Assessment
Case: Tactical Assessment

Problem
- identify strong sniping positions
  - efficiently
  - take into account the scenario
  - also signal absence of good positions

Relevance
- AI in territory based team games
- automated terrain annotation
- team member AI in tactical shooters
Case: Tactical Assessment

Example terrain, top view
(symmetric map)

- Building, 1 door, 2 windows
- Stairs, from water
- Bridge w. 1m / 3ft walls
- Deep water
- Wall, 1m / 3ft
- Wall, 2m / 7ft
Case: Tactical Assessment

example terrain, shots

bottom right: waypoint grid
Case: Tactical Assessment

Approach
1. search among nearby waypoints
2. identify waypoint \( w \) with highest sniping rating towards target
3. check rating \( w \) is good enough
4. if so, move to waypoint \( w \)

and (first)
5. pre-compute sniping rating each waypoint and direction
Case: Tactical Assessment

What makes a good sniping spot?

- inconspicuous
- overlooking distant locations
- hard to reach from target locations
- nearby cover for sniper
- ...

Any relation with waypoints? How to compute these?

- overlooking key locations
- hard to surprise
- focus
- protected flank
- freedom of movement
- target locations lacking cover
Case: Tactical Assessment

Waypoints

- do represent terrain
- can express tactical “ingredients” using:
  - travel time
  - line of sight
  - line of fire
  - distance
  - part of area
  - local environment properties
  - ...

for example: nearby cover for sniper from targets overseen
Case: Tactical Assessment

Idea

\[ \text{tactical value} \left( w, d \right) = \sum k_i \times \text{tactical ingredient}_i \left( w, d \right) \]

Tuning!
It's a heuristic

- \( w \) a waypoint
- \( d \) a direction
Case: Tactical Assessment 10

Same Idea, Lots of Experiments Later

\[
tactical \ value \ (w, d) = \begin{cases} \sum k_i \times local \ property_i (w) \\ + \sum k_j \times group \ membership_j (w) \\ + focus (w, d) \\ \times \sum k_l \times relationship_l (w, d) \end{cases}
\]

① non-directional
② directional
Case: Tactical Assessment

Waypoint local properties
- describe environment
- non-directional

Examples
- light level
- water, low ceiling, power-up, ...
- local movement speed
Case: Tactical Assessment 12

local property: light level
Case: Tactical Assessment

Waypoint group membership
- being part of larger terrain representation
- non-directional

Examples
- room, roof, alley
- base, objective area
Case: Tactical Assessment

Waypoint relations
- (potential) “interaction” or influence between two waypoints
- directional, leveled [0..1]

Examples
- nearby cover from threat
- hard to reach from target
- overlooking key locations
- hard to surprise
Case: Tactical Assessment

Relation: nearby cover from threat

- the amount of nearby cover from threat and nearby positions

\[
\frac{2}{9} = 0.22
\]
Case: Tactical Assessment

nearby cover from seen threats east of position

eastward
Case: Tactical Assessment

Focus

- concentration of relations in certain direction (sphere sector)

Expresses

- flank protection
- view cone
Case: Tactical Assessment

Focus

- implementation depends on
  - relation chosen (line of fire, line of sight, ...)
  - # sectors
  - emphasis on flanks and rear

3D concept

\[
\frac{3 \times N + 2 \times (NE + NW) + 0.5 \times (E + W)}{2 \times (E + W) + 1 \times (SE + S + SW) + 0.5 \times (NE + NW)}
\]
Case: Tactical Assessment

focus east
(E,NE,SE, Eup, Edown)
Combining all waypoint properties

\[ \text{sniping value} (w, d) = \sum_{k} k_i \times \text{local property}_i (w) \times \sum_{j} k_j \times \text{group membership}_j (w) \times \sum_{l} f_{l} \times \text{relationship}_l (w, d) + + \]

17 ingredients, 26 directions, 480 waypoints, 10 seconds PIII/500, et voila... 24KByte of sniping ratings (16bit)
Case: Tactical Assessment

sniping east, using geometry + physics
Case: Tactical Assessment

Results and Process

- waypoint directional sniping quality
  - using “geometry” + actor/weapon physics
- most (key) sniping spots identified
Case: Tactical Assessment

Computed results not perfect
■ formulas / weights not always correct
  ■ easily changed
■ waypoints not always representative
  ■ put in a few more waypoints
■ environment dynamics
  ■ discussed later in this presentation

and
■ game play ignored!
  ■ easily turned into an advantage

Also: visualization errors!
Case: Tactical Assessment

Add reinforcement learning
- analyze sniping behavior / waypoint

- per waypoint (+ direction), reward
  - time spent sniping
    - static, sniper weapon, zoom, shots
  - damage issued while sniping

- per location (+ direction), punish
  - damage received while sniping
positive sniping experience, after 40 rounds, non-directional
Case: Tactical Assessment

- sniping
- east, using
g-ometry + physics + learning
Case: Tactical Assessment

With reinforcement learning

- better sniping ratings
- adaptive AI sniping behavior!!!
  - Al varies position based on performance
Case: Tactical Assessment

We can do even better!

- use game play data in terrain analysis
  - look at (hostile) traffic across terrain
  - look for “kill zones”, “early contact zones”
Case: Tactical Assessment

traffic data (all teams) after 40 rounds
Case: Tactical Assessment

- Sniping
- East, using
- Geometry
- Physics
- Learning
- Tactical
- Game play
- Analysis
Case: Tactical Assessment

Results

- level design
- terrain analysis
- game round
- analyse activity + rerank locations
- reinforcement learning [0.1s]
- sniping activity tracking [0.0s]
- sniper spot analysis [-20 s]
- physics
- waypoints
- line of sight / fire
- paths + travel time
- sniping activity
- traffic + kills...
- sniping ratings
Case: Tactical Assessment

Results

- good sniping ratings
  - all locations, no manual ‘flagging’ required
- adaptive AI
  - responds to in-game success and failure
- tactical AI
  - after game play analysis, sniping ratings become scenario specific
- few in-game resources required
  - 26KB, most of the work is done off-line
Case: Tactical Assessment

Many (similar) Applications

- ambush locations
- strongholds
  - per area per waypoint
- positioning sentry-guns
- base defense
  - use objective in evaluation functions
- generic ‘offensive’ / ‘defensive’ rating
  - see also team maneuvers
- assessing strength of threat position
- ...

William van der Staarren, IGDA Amsterdam chapter meeting, May 11, 2001
Contents

- intro
- case: track and predict opponent
- waypoints, and their use
- case: tactical terrain assessment
- pre-game and in-game analysis
- case: team maneuvers
- issues in implementing terrain reasoning
- conclusions
- GDC do’s and don’ts
Case: Team Maneuver 1

- this case illustrates
  - emergent behavior, taking into account terrain
  - heavily using
    - look up tables
    - evaluation functions
    - pre-computed tactical understanding
  - using waypoints
Case: Team Maneuver 2
Case: Team Maneuver

Problem

- have squad move up / move back
  - tactically
  - as a group
  - using terrain to their advantage
  - in, before and after combat

Relevance

- team AI
- tactical shooters
Case: Team Maneuver 4

how to attack the threats as a team?
Case: Team Maneuver

Approach
- initially, sequentially for each member
  - evaluate nearby waypoints (within $r$) for position improvement
  - claim best position $p$
  - move up to position $p$
- repeat on member initiative
Case: Team Maneuver

For each waypoint evaluate:

- travel time to / from threat, power-up, ...
- nearby cover and flank protection
- tactical rating of position
  - generic tactical rating (similar to sniping)
- min projected distance to team member
  - (spread as seen by threat)
- keep out of friendly line-of-fire (dot product)
- maintain visual contact with enemy
- distance to grenades...
- ...

parameterized for orders and role
Case: Team Maneuver

how to attack the threats as a team?
Case: Team Maneuver 8

evaluate for each waypoint:
- distance to threat
- spread
- nearby cover
- not claimed
- etc.

and claim best position...
Case: Team Maneuver

member1 claims position and will move up.
Case: Team Maneuver

member2 evaluates and picks best position given existing and claimed positions
Case: Team Maneuver

same for member 3

however, in the search radius, the best spot might not be that good (LOF blocked)...

William van der Steen, IGDA Amsterdam chapter meeting, May 31, 2001
upon arrival, member 3 indeed sees a blocked line-of-fire

he quickly executes the same procedure from his new spot...
Case: Team Maneuver

now member3 moves up again to a good position...
in an obstacle rich environment, you’ll get flanking for free...
Case: Team Maneuver 14

Results

- **terrain specific team maneuver**
  - plan new position every time actor wants to move (reload, enemy lost, time out)
  - pretty robust for terrain, team/threat sizes
    - in combat, team size and threat count vary
  - easily parameterized (also from a script)
    - offense, defense, cautious, power-play, objective location, ...

- might also work at higher level...
Case: Team Maneuver 15

**Warning**

- **do use tactical rating of location!**
  - without it, AI too reactive (not pro-active)
    (can be drawn away from stronghold)
  - check ‘best evaluated’ position for
    position improvement

- **inter-member collision avoidance**
  - plan, flock, prioritize… debug
Case: Team Maneuver

Tactical Path Finding

- satisfy more objectives than ‘short’
  - safe (from enemy fire)
  - stealth (from enemy observation)
  - cautious (don’t get caught in weak position)
  - maneuver (offer positions to regroup)

- custom A*
  - heuristics for the cost function
  - heuristics for performance
Case: Team Maneuver

find a safe path to objective, given threat and his line of fire
Case: Team Maneuver

a valid solution
Case: Team Maneuver 19

Approach
- add A* ‘being in the line of fire’ costs

Improvements
- assume spots nearby threat also hostile
  - more robust solution
- use a non-linear function $(x^n, n > 1)$ of exposure time, to prevent long exposure in open areas
Case: Team Maneuver 20

Results

- tactical maneuvering feasible,
- but CPU intensive

Intermediate results improve performance

- waypoints can ‘host’ these results
  - easily used for other reasoning as well
Case: Team Maneuver

Look Up Tables (Intermediate Results)
- per waypoint (and direction)
  - outgoing neighbors
  - (un)protected sectors
    - near and far threats
  - tactical value
  - visible waypoints
  - cover from waypoints
  - travel time to/from neighbor
  - (cached) paths

(un)protected sectors
Contents

- intro
- case: track and predict opponent
- waypoints, and their use
- case: tactical terrain assessment
- pre-game and in-game analysis
- case: team maneuvers
- issues in implementing terrain reasoning
- conclusions
- GDC do’s and don’ts
Implementation Issues

- richer AI
- terrain understanding
- team maneuvers
- adaptive tactics
- CPU intensive
- translated "tactics problem" into well-known problems...

- trade memory for CPU
- caches
- pre-compiled look up tables
- environment dynamics
- improving on $O(N^2)$?
- AI needs + queries
Lookup vs Compute

Path finding / Travel Time
- (tuned) A*: 1,700 paths/s
- lookup: 3,300,000 paths/s

Line-of-Sight/Cover (up to 4 ray traces)
- BSP checks: 25,000 l.o.s./s
- lookup: 4,000,000 l.o.s./s

1 Quakell, PIII/500Mhz, 1K waypoints
Compute vs Lookup

Large Action QII / Counter-Strike map

- ~3MB, 1K waypoints (N)
- dominated by $O(N^2)$ tables
  - line-of-sight matrix
  - path matrix (distance, next node to visit)

- significantly larger terrain
  - more work (data structures, algorithms)

- T&L doesn’t help AI here, Nvidia 😞
Implementation 1

- Resource consumption
- Memory and CPU load
- Dynamic environment
- Reinforcement learning
- Balancing player and AI input
Implementation 2

CPU and Memory Load

- infeasible to do all reasoning on-the-fly
  - too many spots and relations to check
  - many AI actors

- thus pre-computed data required
  - CPU problem turned into memory problem?
  - look for intermediate data that
    - consumes little memory
    - saves lots of CPU
    - is frequently used / on critical path
Implementation 3

Memory Consumption

- terrain reasoning uses more waypoints than navigation
  - to represent terrain properly

- memory consumption dominated by
  - $N \times N$ paths + travel time
  - $N \times N$ lines-of-sight / fire
  - $N$ about 1,000 for larger Half-Life levels

- but terrain is growing
Implementation

Memory Consumption

- > 1,500 waypoints, matrices get large
- path storage
  - hierarchical path finding
  - cache computed paths (AI actor inertia)
- flatten visibility data
  - visible/viewer waypoints p. waypoint

matrix

flat

directional for multiple distances
Implementation 5

Memory Consumption and Performance

■ AI has “planning horizon”
  ■ most reasoning during combat
  ■ most combat decisions are short-term typically 1 to 2 s, depending on game

■ gain performance by storing
  ■ nearby waypoints (< planning horizon)
  ■ all nodes seen by those nearby waypoints
Implementation 6

Dynamic Environment
- doors, vehicles, smoke, lighting, destructible terrain, ...
- invalidate pre-computed results

but
- humans aren’t perfect either
- if exceptions are few, they often can be accommodated
  - often, patching pre-computed tables is more efficient than not using these tables
Implementation 7

Reinforcement Learning

- value human game play more than AI
  - player is more creative
  - typically more AI than humans present
  - AI may hone into its flaws otherwise

- visualization
  - terrain reasoning is heuristics
  - set-up feedback facilities early on
  - preferably in game engine (overlays)
Contents

- intro
- case: track and predict opponent
- waypoints, and their use
- case: tactical terrain assessment
- pre-game and in-game analysis
- case: team maneuvers
- issues in implementing terrain reasoning
- conclusions

- GDC do’s and don’ts
Conclusions 1

3D level design
theater to shape interaction
every spot has a (tactical) purpose
stronghold, base, avenues,...

doable
+ “documented”

richer reaction
and interaction
more realism
more variation

AI running around
Conclusions 2

Terrain (tactical) concepts such as

- terrain analysis
- situational awareness
- team maneuvers, ...
- threat assessment
- team positioning (regroup, defend, ...)

- can be expressed as evaluation functions per location
- as heuristics, but with...
  - clear relation to original concept

partially adaptive!
partially pre-computed!
Conclusions 3

Analysis 10% : Experiment 40% : Tune 50%

- simple ideas: no rocket science
  - (thus?) little literature available 😞
- experiment a lot
  - see what (emergently) solves your problem
  - see which imperfections you can away with
- tune! tune! tune!
  - too much terrain and more to come
  - measure and visualize first, then modify
  - design data structures after AI (query) patterns
- yep, this will take you time

start early, repeat
References + Suggested Reading

See

- “links collection” at www.cgf-ai.com
Credits

- reviewers (proceedings, presentation)
  - Jan Paul van Waveren
  - Doug Reece
  - Tobias Heimann
  - Markus Klinge

- screen shots from Quake II
  - by Id Software
  - content from mod Action Quake2
    - action.telefragged.com
    - www.botepidemic.com/aid/cgfeatures.shtml
Questions From the GDC Audience...

- Waypoint distribution?
- Waypoint placement?
- Do the algorithms run in real-time?
Questions?

Q Do the waypoints need to be distributed as a uniform grid?

A Most algorithms presented assume a more or less uniform distribution of waypoints across the terrain (see ‘focus’). However, the algorithm is largely robust to deviations from the uniform distribution. It makes sense to cover outdoor feature-less terrain with fewer waypoints, while increasing the waypoint density near obstacles.
Questions?

Q Are the waypoints automatically placed?

A The waypoints are placed semi-automatically. The process is not fully automatic (as, for example, Quake III Arena), but requires a player to visit all locations in the game. Waypoints are spawned automatically when the player visits location not visited before.

Note that the algorithms are independent of the way the waypoints are positioned.
Questions?

Q Do the algorithms run in real-time?

A Except for the sniping spot analysis, all algorithms run fine ‘real-time’ in the game (most of them were developed to run on a Celeron300, with 16+ AI NPC’s in the game).

The A* tactical path finding in general is the most expensive algorithm to run.
William’s GDC2001 Experiences

- Boy’s dream come true
- Good time with like-minded people nerds
- Roundtables + AI Programmer’s Dinner

My trip preparation was just slide preparation
- so, some do’s and don’ts
William’s GDC2001 Experiences

GDC Visit “Do’s”

- arrange meetings in advance
  - impossible to find & contact people at GDC
- know whom to speak to

- wear T-shirt with company name, logo, ...
  - you want to be found as well

- present real content, “how to” stuff
  - apparently everybody can do the flashy stuff
William’s GDC2001 Experiences

GDC Visit “Don’ts”

- don’t forget business cards, demo CDs, or slide handouts
  - help your contacts not to forget about you

- don’t skip the recommended $$$ hotels
  - build contacts during breakfast and dinner

- don’t believe your cell phone subscription really works in the US