The Official Counter-Strike Bot
Michael Booth, CEO
Turtle Rock Studios
mike@turtlerockstudios.com

What is Counter-Strike

- World’s #1 online action game
- Terrorists vs Counter-Terrorists
- Scenarios
  - Defuse the Bomb
  - Rescue the Hostages
  - Escort the VIP
- More than “twitch” reflexes
  - Teamwork and tactics are essential
  - Stealth is important
  - Semi-realistic weapon properties (recoil, bullet spray)
- Online multiplayer only (until now)
The Design Goals of the Official Counter-strike Bot

• Bring the “Counter-Strike Experience” to solo players
• Be fun for both new and veteran players
• Play all aspects of Counter-Strike well
• Behave in a believably human manner
• Be simple to use
• Be customizable

Things the CS Bot Must Do

• “Understand” a Complex 3D Environment
• Move Around in the Environment
• Perceive Objects and Events
• Communicate with Teammates
• Decide What to Do Next
• Be Proficient in Combat
• Behave in a Human-like Manner
“Understanding” a Complex 3D Environment

- Representing the Environment
  - Any sort of reasoning requires a representation on which to work
  - Without reasoning, a bot can only react (insect-like)

- Waypoints
  - A Waypoint defines a useful position in the environment
  - Waypoints are connected to each other, creating a searchable network

- The Problem with Waypoints
  - Inherently one-dimensional
  - Only safe route is directly along the line between two waypoints
  - “Thick” waypoint paths help, but are still inadequate
Simple Waypoint Example

“Understanding” a Complex 3D Environment

- A Better Solution: Navigation Meshes
- Navigation Areas represent 2D “walkable” areas
  - Paths can pass through the area anywhere
  - Useful for randomizing positions
- Areas are connected along their edges to other Areas
  - Bi-directional
  - One-way
  - Ladder
- Areas may have attributes
  - Jump
  - Crouch
  - “Danger”
  - Important Scenario locations (bomb site, hostage rescue)
Simple Navigation Area Example

Navigation Areas: Dust
Navigation Areas: Dust

[Image of a dusty navigation area with highlighted paths]
Navigation Areas: Dust

Moving Around in the Environment

- Dealing with Features of the Environment
  - Corners/Irregular Edges/Small Obstacles
  - Jump-up Ledges
  - Jump-over Gaps
  - One-way Drop-offs
  - Doors
  - Ladders
  - Vents (crouch areas)
  - Breakables (windows, etc)
Ledge and Gap Jumping Navigation Mesh

Ledge and Gap Jumping Example
Movement Hierarchy

- Movement Planning & Pathfinding
- Path Following
- Reactive Obstacle Avoidance Behavior
- Generation of Movement Commands
- Standard Player Movement Physics
- Actual Player Position Changes

Path Following
Reactive Obstacle Avoidance

Navigation Example: Italy
Navigation Example: Office

Moving Around in the Environment

• Pathfinding
  – A* Cost Function
    • Increased cost for crossing Jump and/or Crouch areas and Ladders
    • Add “Danger” costs depending on “Safest” or “Fastest” route
Moving Around in the Environment

- Getting Stuck
  - Causes
    - Small and/or irregular obstacles
    - Ladders
    - Falling off the path
    - Friends blocking your way
  - Detection
    - Watch average velocity over a short window of time
  - Getting Un-Stuck
    - Random "wiggle"
    - Add random jump after a short duration

Perceiving the Environment

- Directing Attention (Looking around)
  - Currently selected Victim
  - Hiding Spots
  - Last known Enemy location
  - Approach Points
  - Towards Enemy noises
  - Navigating Ladders

- Hearing noises
  - Randomizing location
  - Prioritizing

- React to other “Interesting” Game Events
  - Scenario announcements (“The bomb has been planted”)
  - Injuries
  - Kills
  - Bullet impacts
  - Windows breaking
Hiding Spots: Dust

Approach Points: Dust
Controlling the View

• Viewing is Independent of Movement
  – Required for a realistic Counter-Strike performance
  – Allows visual searching while moving
  – Allows tactical movement during combat
  – Allows firing at enemies while retreating

• View Control is Critical
  – Bots can only “see” what is in their field of view

Controlling the View: View Behavior Hierarchy
Controlling the View: View Physics

• Use accelerations
  – Keeps motion smooth and continuous

• Spring/damper model
  – Increase spring stiffness when aiming

\[ \omega' = (k \alpha - d \omega) \Delta T \]

- \( \omega \) = angular velocity
- \( k \) = spring stiffness
- \( \alpha \) = angular acceleration
- \( d \) = damping coefficient
- \( \Delta T \) = duration of timestep

Looking Around Example
Deciding What to Do

- Scenario objectives
- Danger
- Morale
- Teamwork
Deciding What to Do: Danger

- During the game, when a teammate dies, an amount of "danger" is added to all nearby Navigation Areas
- "Danger" slowly decays over time
- "Danger" is used in the A* cost function when building a path to a location
- An individual bot’s "personality" modulates how much "Danger" costs
- Causes bots to choose different routes each round

Deciding What to Do: Morale

- Each bot has a "Morale" value
- Morale is increased when the bot:
  - Kills an enemy
  - Completes an objective
  - Is on the winning team
- Morale is decreased when the bot:
  - Is killed
  - Is on the losing team
- Morale is used when determining whether to "rush" or "camp"
Combat

- How to aim
- Weapon proficiency
- Dealing with Danger
- Hiding/Sniping
- Retreating
- Stealth
- Following a Leader

Combat: How to Aim

- Select point P on victim
- Add Aim Offset, resulting in P’
  - Periodically generate new random Aim Offset based on bot’s “skill”
- Add View Drift, resulting in P”
- Apply angular forces to orient view towards P”
Combat: Aiming Example
(Easy Difficulty)

Combat: Aiming Example
(Hard Difficulty)
**Combat: Weapon Proficiency**

- Rules that capture the subtleties of each kind of weapon
  - Burst firing with a Rifle when victim is far away
  - Switching to a pistol instead of reloading primary weapon
  - Switching to a pistol if using a sniper rifle and victim is very close
  - How and when to zoom in if weapon has a scope
  - Switching to a pistol if using a shotgun and victim is very far away
  - Aiming for the head, unless using certain sniper rifles, shotguns, or the knife
  - Knives are melee weapons, not ranged
  - Have to throw grenades far ahead
  - Understanding how to attack against and defend with the Tactical Shield

**Combat: Hiding / Sniping**

- When a bot decides to hide, guard a location, or snipe, it picks a nearby Hiding Spot in the Navigation Mesh
- All Hiding Spots provide good cover
- Some Hiding Spots are flagged as Sniping Spots, due to their long lines of sight
- Depending on current behavior, the maximum range for Hiding Spot selection varies
- Must do Breadth First Search through Nav-Mesh when finding Hiding Spots – Linear distance to spot can be very misleading
- Must take care not to select a Hiding Spot that is already occupied
Combat: Retreating

• If a bot decides to retreat, it selects a nearby Hiding Spot that:
  – No known enemies have line of sight to
  – Is closer to friends than any enemies

• If all Hiding Spots are visible to enemies, one visible to the fewest is selected
**Combat: Stealth**

- If a bot is alone or has very few teammates left and hears an enemy noise nearby, it will investigate while walking silently ("sneaking")
- Reports situation to its team ("I hear something")
- Players are often surprised by sneaking bots

**Combat: Following a Leader**

- Periodic re-path if leader is moving
- Walk if leader is walking, run if leader is running
- If leader is hiding, take up covering position nearby
- Following behind the leader is straightforward
- Much more difficult when "taking the point"
  - Must predict where leader is going in a generalized way ("to the house", "to bombsite b")
Communication and Teamwork

• Reacting to Friends Actions
  – “Enemy spotted”
  – “Need backup”
  – Friend firing at target
  – Friend death

Talking to Teammates

• What to say
  – Expose internal state
    • “I’m going to guard bombsite B”
    • “I’m going to plant the bomb”
  – Report situation
    • Location where enemy was spotted
    • Number of enemies remaining
    • Status of scenario (“They dropped the bomb”)

• When to say it
  – Be careful to not talk over friends
  – Don’t repeat yourself, or what your teammates say
  – Don’t talk too much, or too often

• How to say it
  – Have as many variations of each phrase as possible
  – Never emit the same phrase twice in a row
The Importance of Being Human

• Player perception of a “fair fight”

• Creates behavior variation among bots
  – Aggression
  – Skill
  – Teamwork
  – Reaction Times
  – Morale

• Results in a more complete game experience
  – Getting the drop on a bot
  – Tactics remain important
  – “Out-thinking” a bot

Simple to Use: User Interface
Simple to Use: Automated Map Learning

- Important due to the many community-created maps that exist
- When a bot is added to a new map, it spends a few minutes learning it and generates a Navigation file
- Learning samples the map, and aggregates the samples into Nav Areas via a greedy algorithm
- Automatically finds Hiding Spots and Approach Points
Customization

- Adding bots by name
- Editing bot “personalities”
  - Behavior attributes
  - Weapon preferences
  - Specifying different bot “skins”
  - Specifying different bot voices

Making it Fun

- Difficulty levels
- Design behaviors for exciting “moments”
- Respect the Player
Making it Fun: Difficulty levels

- **Easy**
  - Poor reaction times
  - Terrible aim
  - Substantial additional delay before opening fire on victim
  - Poor weapon proficiency
  - Inferior Weapon selection
- **Normal**
  - Medium reaction times
  - Medium aim
  - Minor additional attack delay
  - Mix of weapon proficiencies
- **Hard**
  - Good reaction times
  - Good aim
  - No additional attack delay
  - Complete weapon proficiency
- **Expert**
  - Very good (but still human) reaction times
  - Excellent aim

Making it Fun: Exciting “Moments”

- Not avoiding Flashbangs well
- Retreating
- “Focusing in”
- Using Stealth
Making it Fun: Respect the Player

• Make the player feel smart, skilled, and in charge
  – Bots refer to Players as “Sir” or “Commander”
  – Bots occasionally congratulate the Player on a kill (“Nice shot, sir”)
  – By default, bots defer key scenario objective to human Players

Code Philosophy

• Solid AI takes time
  – The last 10% will take 90% of the time
  – Navigation is hell

• Behavior code complexity
  – Several concurrent state machines
  – Complexity grows geometrically
  – Clean code structure, verbose naming conventions, and lots of comments
  – State machine encapsulation
    • MyState.OnEnter()
    • MyState.OnUpdate()
    • MyState.OnExit()
    • SetState( State *newState )
  – Use algorithms because they are useful, not because they are cool
Performance is Key

- An AI Engineer is a “Virtual Puppeteer”
  - Players do not care about cool algorithms
  - It’s all about putting on a good show
  - Suspension of Disbelief
    - Players will rationalize most behaviors
    - Blatantly mindless behavior breaks the illusion
  - You are successful when nobody notices

The CS Bot in Action