Official YaPB Documentation

Release latest

unknown

Apr 16, 2024

BOT DOCUMENTATION

YaPB is an AI opponent for Counter-Strike which allows you to play good old Counter-Strike without connecting to any server, or fill your server with AI-controlled players.

If you have time and interest you can contribute to documentation on GitHub repository.

CHAPTER

ONE

INTRODUCTION

1.1 What is YaPB?

YaPB stands for Yet another Ping of Death Bot. It's a deriviate of a Counter-Strike famous bot - **PODBot**, based on it's source code release by Count-Floyd back in 2003. It's adding computer-controlled (AI) players to Counter-Strike games based on GoldSource engine.

1.2 Why is it called YaPB?

It's called "Yet another PODBot" because when fork was started, there was so many other PODBot clones floating around like **PODBot MM**, **e[POD]bot**, **IvPBot**, **POXBot** etc. So the reason was to save the original name, and clarify that bot is derived from the original PODBot.

And POD is an abbreviation for **Ping of Death**. It's called that way because it was derived from the **High Ping Bastard Bot** Botman was doing. And if you look at the Bots Ping they always have a Ping of 5 in the days of Counter-Strike 1.5 and the very first generation of bots did Headshots all the time.

1.3 What are Bots able to do?

The Bots will automatically know the goals of the map. Counter-Terrorists will go for hostages or defuse the bomb, Terrorists will plant the bomb. VIP Bots will try to reach the Rescue Points. The goal selection is done dynamically and will be affected by several factors like Personality, Health, Teammates nearby & Items the Bot is carrying. The Bots will also try to support teammates and take cover when they think it's needed. They also can use some elevators.

1.4 What mods are supported?

YaPB currently supports only Counter-Strike (since Beta 6.5 to 1.6) and Condition Zero mods for Half-Life.

- Supported engines:
  - Official Valve's GoldSrc Engine on Windows, Linux and macOS.
  - Flying with Gauss Xash3D Engine on Windows, Linux, macOS and Android.
  - ReHLDS & ReGameDLL on Windows and Linux.
  - WON version of Valve's GoldSrc Engine on Windows, Linux. Minimum engine version is 1.1.0.4.

YaPB supports running on a listenserver (when the players hosts game itself), as well as on HLDS (dedicated server).

1.5 Why does YaPB need waypoints?

Is there a Bot who is doing great without any kind of navigational help like waypoints, wayzones or navmesh? It's almost certain you won't find any. Actually, at the time of original development, waypointed navigation was the most widely-used and simple method. Bots are currently navgiating flawlessly, so changing it to navmesh or any other stuff is not planned. YaPB maintains big list of waypoints on the github repository which is constantly replenished, so probably we already have the waypoint for your map...

If you would like a waypoint for your map, you can do a issue-request on the github repository. But it's better to make waypoints yourself than to wait for someone to make waypoints for you.

1.6 What types of maps are supported?

- Supported game scenarios:
  - Aim Training (aim_* maps).
  - Escape (es_* maps).
  - Knife Arena (ka_* maps).
  - Deathmatch (dm_* maps).
  - Fight Yard (fy_* maps).
  - Gun Game (gg_* maps).
  - Grenade Wars (he_* maps).
  - Prepared Assault (pa_* maps)
  - VIP Assasination (as_* maps).
  - Sniper Wars (awp_* maps).
  - Hostage Rescue (cs_* maps).
  - Bomb Plant/Defuse (de_* maps).
  - Hostage Rescue/Bomb Plant/Defuse (csde_* maps).
  - Speed Strike (sp_* maps).

1.7 What mods are supported for Counter-Strike?

YaPB officially supports CSDM and CSDM FFA mods for Counter-Strike. Other mods should add their own support for bots.

CHAPTER

TWO

INSTALLATION

Table of Contents

2.1 Brief information

2.1.1 Before you begin

Unlike many other PODBot flavours, YaPB is able to run as metamod plugin as well as standalone dll for every platform it supports (except Android) without changing so/dll/dylib.

Important: If you have some mods like AMX Mod X installed, you should consider installing bots as a metamod plugin, not as standalone dll. In other situations there is **no** requirement to install metamod before installing YaPB.

Installation docs only cover installing bot on genuine Steam version of Counter-Strike. Assuming that if you run something special, you know how to install bot as standalone dll or as metamod plugin.

Note: Only 32-bit binary is available, since Valve has dropped support for 64-bit HLDS.

2.2 Installing bot on Windows

YaPB supports installing both on listen and dedicated server under Windows. Many people are using it as their main operating system, so the bot has a simple installer, that could assist you in adding bots to game. If you don't want to install manually, please use an installer.

2.2.1 Installing using setup program

- Download the latest YaPB bot installer from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-windows.exe`)
- Run program.
- Press `Browse` and point to where `hl.exe/hlds.exe` is located.
- Press `Install` to install bot.
- You're done!

Important: Installer program automatically detects whether you use metamod, and will patch `plugins.ini` file automatically for you, adding entry to point to `yapb.dll`. If there is no metamod, installer will patch the `liblist.gam` file to point to `yapb.dll`

Caution: Installer has a feature for detecting Counter-Strike game folders. It's looks up all directories where selected `hl.exe/hlds.exe` is located and checks the library for containing export symbols `weapon_ak47` and `weapon_m4a1`, and if so assuming that this folder contains some Counter-Strike version, it will install bot to this directory automatically.

2.2.2 Installing without metamod

Assuming that your Counter-Strike is located at: `C:\Steam\steamapps\common\Half-Life\cstrike`.

Note: If you're installing bot for Condition Zero, then install it to `czero` folder instead.

- Download the latest `ZIP` YaPB package from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-windows.zip`)
- Unzip the downloaded package to `C:\Steam\steamapps\common\Half-Life\cstrike` directory.
- Open a file called `liblist.gam` in Notepad or any other text editor. (located in `C:\Steam\steamapps\common\Half-Life\cstrike`.)
- Locate the entry `gamedll`. It should point to `dlls\mp.dll`.
- Change entry from `gamedll "dlls\mp.dll"` to `gamedll "addons\yapb\bin\yapb.dll"`.
- Save the changes.

- You're done!

2.2.3 Installing as metamod plugin

Assuming that your Counter-Strike is located at: `C:\Steam\steamapps\common\Half-Life\cstrike`.

Note: If you're installing bot for Condition Zero, then install it to `czero` folder instead.

- Download the latest `ZIP` YaPB package from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-windows.zip`)
- Unzip the downloaded package to `C:\Steam\steamapps\common\Half-Life\cstrike` directory.
- Open a file called `plugins.ini` in Notepad or any other text editor. (located in `C:\Steam\steamapps\common\Half-Life\cstrike\addons\metamod`.)
- Append the following entry: `win32 addons/yapb/bin/yapb.dll`.
- Save the changes.
- You're done!

2.3 Installing bot on Linux

YaPB supports installing both on listen and dedicated server under Linux. No installer provided for the Linux.

2.3.1 Installing without metamod

Assuming that your Counter-Strike is located at: `/usr/Steam/steamapps/common/Half-Life/cstrike`.

Note: If you're installing bot for Condition Zero, then install it to `czero` folder instead.

- Download the latest `TAR.XZ` YaPB package from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-linux.tar.xz`)
- Extract the downloaded package to `/usr/Steam/steamapps/common/Half-Life/cstrike` directory.
- Open a file called `liblist.gam` in any text editor. (located in `/usr/Steam/steamapps/common/Half-Life/cstrike`)
- Locate the entry `gamedll_linux`. It should point to `dlls/cs.so`.
- Change entry from `gamedll_linux "dlls/cs.so"` to `gamedll_linux "addons/yapb/bin/yapb.so"`.
- Save the changes.
- You're done!

2.3.2 Installing as metamod plugin

Assuming that your Counter-Strike is located at: `/usr/Steam/steamapps/common/Half-Life/cstrike`.

Note: If you're installing bot for Condition Zero, then install it to `czero` folder instead.

- Download the latest `TAR.XZ` YaPB package from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-linux.tar.xz`)
- Extract the downloaded package to `/usr/Steam/steamapps/common/Half-Life/cstrike` directory.
- Open a file called `plugins.ini` in any text editor. (located in `/usr/steam/steamapps/common/half-life/cstrike/addons/metamod`.)
- Append the following entry: `linux addons/yapb/bin/yapb.so`.
- Save the changes.
- You're done!

2.4 Installing bot on macOS

YaPB can be installed on listen servers only if you use macOS.

Note: macOS support is deprecated. YaPB binary moved to extras package and is located in the `darwin-x86` folder.

Note: There is unofficial binary called `hlds_osx` provided by AlliedModders LLC allowing you to run HLDS under macOS. You can try download it here.

Note: Valve didn't update GoldSrc games to support macOS Catalina, and Apple has dropped 32-bit support, so Counter-Strike isn't playable on latest macOS releases. YaPB still provides binaries for pre-Catalina users. As soon as Valve (if ever) release Counter-Strike for macOS Catalina, YaPB 64-bit binaries will be available.

2.4.1 Installing without metamod

Assuming that your Counter-Strike is located at: `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike`.

Note: If you're installing bot for Condition Zero, then install it to `czero` folder instead.

- Download the latest YaPB package from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-windows.zip` or `yapb-4.4.957-linux.tar.xz`)
- Download the latest extras package from https://yapb.jeefo.net/latest (example: ``yapb-4.4.957-extras.zip)
- Unzip the downloaded YaPB package to `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike` directory.

- Unzip the `yapb.dylib` binary from the `darwin-x86` folder in the downloaded extras package to `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike/addons/yapb/bin` directory.
- Open a file called `liblist.gam` in any text editor. (located in `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike`.)
- Locate the entry `gamedll_osx`. It should point to `dlls/cs.dylib`.
- Change entry from `gamedll_osx "dlls/cs.dylib"` to `gamedll_osx "addons/yapb/bin/yapb.dylib"`.
- Save the changes.
- You're done!

2.4.2 Installing as metamod plugin

Assuming that your Counter-Strike is located at: `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike`.

Note: If you're installing bot for Condition Zero, then install it to `czero` folder instead.

- Download the latest YaPB package from https://yapb.jeefo.net/latest (example: `yapb-4.4.957-windows.zip` or `yapb-4.4.957-linux.tar.xz`)
- Download the latest extras package from https://yapb.jeefo.net/latest (example: ``yapb-4.4.957-extras.zip)
- Unzip the downloaded YaPB package to `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike` directory.
- Unzip the `yapb.dylib` binary from the `darwin-x86` folder in the downloaded extras package to `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike/addons/yapb/bin` directory.
- Open a file called `plugins.ini` in any text editor. (located in `/Users/user/Library/Application Support/Steam/steamapps/common/Half-Life/cstrike/addons/metamod`)
- Append the following entry: `osx addons/yapb/bin/yapb.dylib`.
- Save the changes.
- You're done!

CHAPTER

THREE

CONFIGURING THE BOT

Bot CVARS are located inside yapb.cfg and parsed on every level change. So new config will be applied on map change or by issuing `exec addons/yapb/conf/yapb.cfg` inside the server console.

CVARs can be changed on the fly by typing them into the console.

3.1 Bot CVARS

3.1.1 yb_attack_monsters

Allows or disallows bots to attack monsters.

Useful for mods like Halloween Mod.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.2 yb_autokill_delay

Specifies amount of time in seconds when bots will be killed if no humans left alive.

Minimum value is `0`, Maximum value is `90`, Default value is `0`.

3.1.3 yb_autovacate

If not zero, bots will automatically leave to make room for human players, when they join the server.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.4 yb_autovacate_keep_slots

How many slots autovacate feature should keep for human players

Minimum value is `1`, Maximum value is `8`, Default value is `1`.

3.1.5 yb_avoid_grenades

Allows bots to partially avoid grenades.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.6 yb_bind_menu_key

Binds the specified key to open a bot menu.

Default value is =.

3.1.7 yb_botbuy

If not zero, bots will be able to buy weapons and inventory.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.8 yb_botskin_t

Specifies the bots wanted skin for Terrorists team.

Allowed values:

- `0` - Any class
- `1` - Phoenix Connexion
- `2` - Elite Crew
- `3` - Arctic Avengers
- `4` - Guerilla Warfare
- `5` - Midwest Militia **(Condition Zero only!)**

Minimum value is `0`, Maximum value is `5`, Default value is `0`.

3.1.9 yb_botskin_ct

Specifies the bots wanted skin for Counter-Terrorists team.

Allowed values:

- `0` - Any class
- `1` - Seal Team 6
- `2` - GSG-9
- `3` - SAS
- `4` - GIGN
- `5` - Spetsnaz **(Condition Zero only!)**

Minimum value is `0`, Maximum value is `5`, Default value is `0`.

3.1.10 yb_breakable_health_limit

Specifies the maximum health of breakable object, that bot will consider to destroy.

Minimum value is `1`, Maximum value is `3000`, Default value is `500`.

3.1.11 yb_camping_allowed

If not zero, bots will try to pickup camp points as their goals, and will camp there for some time based on their personality.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.12 yb_camping_time_min

Lower bound of time from which time for camping is calculated.

Minimum value is `5`, Maximum value is `90`, Default value is `15`.

3.1.13 yb_camping_time_max

Upper bound of time until which time for camping is calculated.

Minimum value is `15`, Maximum value is `120`, Default value is `45`.

3.1.14 yb_chat

If not zero, bots will be able to chat to each other and players while they are dead.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.15 yb_chat_percent

Bot chances to send random dead chat when killed.

Minimum value is `0`, Maximum value is `100`, Default value is `30`.

3.1.16 yb_check_darkness

Allows or disallows bot to check environment for darkness, thus allows or not to use flashlights or NVG.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.17 yb_check_enemy_invincibility

Enables or disables checking enemy invincibility. Useful for some mods.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.18 yb_check_enemy_rendering

Allows to check enemy rendering before taking a victim. This is useful to enable when you plays CSDM mod with spawn protection enabled. Bots will not try to select just-spawned players, as they are not vulnerable.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.19 yb_csdm_mode

Enables or disables CSDM / FFA mode for bots.

- If set to `0`, CSDM / FFA mode is auto-detected.
- If set to `1`, CSDM mode is enabled, but FFA is disabled.
- If set to `2`, CSDM and FFA mode is enabled.
- If set to `3`, CSDM and FFA mode is disabled.

Minimum value is `0`, Maximum value is `3`, Default value is `0`.

3.1.20 yb_chatter_path

Points to location where chatter (from official csbot for example) is located.

Minimum value is `-`, Maximum value is `-`, Default value is `sound/radio/bot`.

3.1.21 yb_count_players_for_fakeping

Take players' pings into account when calculating the average ping for bots. If not, a random ping will be chosen for bots.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.22 yb_debug

If not zero, enables useful messages about bot states. Not required for end users.

Minimum value is `0`, Maximum value is `4`, Default value is `0`.

3.1.23 yb_debug_goal

Forces all alive bots to build path and go to the specified here graph node.

Minimum value is `-1`, Maximum value is `2048`, Default value is `-1`.

3.1.24 yb_destroy_breakables_around

Allows bots to destroy breakables around him, even without touching with them.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.25 yb_difficulty

Specifies the difficulty of all bots. Changing at runtime will affect already created bots.

List of bot difficulties:

- `0` - Newbie,
- `1` - Average,
- `2` - Normal,
- `3` - Professional,
- `4` - Godlike.

Minimum value is `0`, Maximum value is `4`, Default value is `4`.

3.1.26 yb_difficulty_auto_balance_interval

Interval in which bots will balance their difficulty.

Minimum value is `30`, Maximum value is `240`, Default value is `30`.

3.1.27 yb_difficulty_min

Lower bound of random difficulty on bot creation. Only affects newly created bots. `-1` means yb_difficulty only used.

Minimum value is `-1`, Maximum value is `4`, Default value is `-1`.

3.1.28 yb_difficulty_max

Upper bound of random difficulty on bot creation. Only affects newly created bots. `-1` means yb_difficulty only used.

Minimum value is `-1`, Maximum value is `4`, Default value is `-1`.

3.1.29 yb_difficulty_auto

Enables each bot balances own difficulty based kd-ratio of team.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.30 yb_display_welcome_text

Specifies if the bot dll will display welcome text when adding bots.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.31 yb_display_menu_text

Enables or disables display menu text, when players asks for menu. Useful only for Android.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.32 yb_economics_disrespect_percent

Allows bots to ignore the economics and buy weapons with disrespect of it.

Minimum value is `0`, Maximum value is `100`, Default value is `25`.

3.1.33 yb_economics_rounds

If not zero, bots will use "team" economics, if more than 70% of players don't have money to buy preferred weapon no-one buy anything to save money for the next round. This usually causes bots running with default pistols on first round.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.34 yb_enable_fake_steamids

Allows or disallows bots to return fake Steam ID.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.35 yb_enable_query_hook

Enables fake server queries response, that shows bots as real players in server browser.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.36 yb_freeze_bots

If not zero, bots think function is paused and bots stays at the place where they were been before activating these cvar.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.37 yb_graph_analyze_auto_save

Automatically saves the analysis results to a graph file. And adds bots again.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.38 yb_graph_analyze_auto_start

Starts the map autoanalyzer if the graph is not present in the local storage or database.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.39 yb_graph_analyze_clean_paths_on_finish

Specifies if analyzer should clean the unnecessary paths upon finishing.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.40 yb_graph_analyze_distance

Specifies the minimum distance to keep nodes from each other.

Minimum value is `42`, Maximum value is `128`, Default value is `64`.

3.1.41 yb_graph_analyze_fps

Specifies the FPS at which analyzer process is running. This keeps game from freezing during analyzing.

Minimum value is `25`, Maximum value is `99`, Default value is `30`.

3.1.42 yb_graph_analyze_mark_goals_on_finish

Specifies if analyzer should mark nodes as map goals automatically upon finish.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.43 yb_graph_analyze_max_jump_height

Specifies the max jump height to test if next node will be unreachable.

Minimum value is `44`, Maximum value is `64`, Default value is `44`.

3.1.44 yb_graph_analyze_optimize_nodes_on_finish

Specifies if analyzer should merge some near-placed nodes with much of connections together.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.45 yb_graph_auto_collect_db

Allows the bot to scan the `graph` folder and upload every single `.graph` file to the database if they are not there. This is done in a separate thread and does not block the server process.

Note: It works only at server startup, not at map change. Also it does not work on the currently started map.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.46 yb_graph_auto_save_count

Every N graph nodes placed on map, the graph will be saved automatically (without checks).

If you want to disable autosave, set this cvar to `0`.

Minimum value is `0`, Maximum value is `2048`, Default value is `15`.

3.1.47 yb_graph_draw_distance

Maximum distance to draw graph nodes from editor viewport.

Minimum value is `64`, Maximum value is `3072`, Default value is `400`.

3.1.48 yb_graph_fixcamp

Specifies whether bot should not 'fix' camp directions of camp waypoints when loading old PWF format.

Note: This option was made to fix camp directions when using waypoints from PODBot 2.5, old YaPB versions or other PODBot clones that do not allow to set a horizontal camp direction. If you are using pwf waypoints from PODBot 3.0 this cvar should be disabled, as it can break the directions of the camp nodes.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.49 yb_graph_url

Specifies the host where graph database is located. They must be in `/graph` path on the server. Set to empty, if you don't want downloads

Allowed values: Valid DNS hostname with HTTP server listening on port 80. Default value is `yapb.jeefo.net`.

3.1.50 yb_graph_url_upload

Specifies the URL to which bots will try to upload the graph file to database.

Default value is `yapb.jeefo.net/upload`.

3.1.51 yb_grenadier_mode

If enabled, bots will not apply throwing condition on grenades.

Minimum value is `0`, Maximum value is 1, Default value is `0`.

3.1.52 yb_has_team_semiclip

When enabled, bots will not try to avoid teammates on their way. Assuming that some of the semiclip plugins are in use.

Minimum value is `0`, Maximum value is 1, Default value is `0`.

3.1.53 yb_ignore_cvars_on_changelevel

Comma separated list of bot cvars to ignore on changelevel.

Bots reads `yapb.cfg` every changelevel, and the values of bot cvars are overwritten with those located in config. This cvar allows server admin to ignore values of specified here cvars from `yapb.cfg` if they were changed by hand from server console.

For example: Server started with `yb_quota` set to `10` in `yapb.cfg` and `yb_quota` is specified in `yb_ignore_cvars_on_changelevel`. Time passed, server admin decide that `yb_quota` should be set to `12` and set it via server console. Next time server will change map, value of `yb_quota` will not be changed while reading `yapb.cfg` and will stay `12`.

If you want to disable ignoring reading cvars from `yapb.cfg`, then leave the value empty `yb_ignore_cvars_on_changelevel ""` in `yapb.cfg`

Minimum value is -, Maximum value is -, Default value is `yb_quota,yb_autovacate`.

3.1.54 yb_ignore_enemies

If not zero, bots will run all over the map, and doing goals but will not search for enemies.

Minimum value is `0`, Maximum value is 1, Default value is `0`.

3.1.55 yb_ignore_enemies_after_spawn_time

Make bots ignore enemies for a specified here time in seconds on new round. Useful for Zombie Plague mods.

Minimum value is `0`, Maximum value is `540`, Default value is `0`.

3.1.56 yb_ignore_map_prefix_game_mode

If enabled, bots will not apply game modes based on map name prefix (**fy_** and **ka_** specifically).

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.57 yb_ignore_objectives

Allows or disallows bots to do map objectives, i.e. plant/defuse bombs, and saves hostages.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.58 yb_jasonmode

If not zero, bots will use only knives while fighting against enemies. This also disabling buying.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.59 yb_join_after_player

If not zero, bots will join the server only when some human player already joined the team.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.60 yb_join_team

Forces all bots to join team specified in this cvars.

Valid values: `ct`, `t`, `any`, Default value is `any`.

3.1.61 yb_join_delay

Specifies after how many seconds bots should start to join the game after the changelevel.

Minimum value is `0`, Maximum value is `30`, Default value is `5`.

3.1.62 yb_kick_after_player_connect

Kick the bot immediately when a human player joins the server (yb_autovacate must be enabled).

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.63 yb_language

Sets the bot language for menus, names, chat and messages.

Valid values: `ru`, `en`, `de`, Default value is `en`.

3.1.64 yb_logger_disable_logfile

Disables logger to write anything to log file. Just spew content to the console.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.65 yb_max_nodes_for_predict

Maximum number for path length, to predict the enemy.

Minimum value is `15`, Maximum value is `256`, Default value is `25`.

3.1.66 yb_name_prefix

This cvar contains a string that will be prepended to every added bot name. Something like a clantag.

By default this value is not set.

3.1.67 yb_object_destroy_radius

The radius on which bot destroy breakables around it, when not touching with them.

Minimum value is `64`, Maximum value is `1024`, Default value is `400`.

3.1.68 yb_object_pickup_radius

The radius on which bot searches world for new objects, items, and weapons.

Minimum value is `64`, Maximum value is `1024`, Default value is `450`.

3.1.69 yb_password_key

Specifies the password key for `setinfo` command, to gain remote control to `yb` command and bot menus.

Default value is: `_ybpw`.

3.1.70 yb_password

Specifies the actual password for `setinfo` command, to gain remote control to `yb` command and bot menus.

To gain access to bot command remotely, user should open console and enter `setinfo key password`, when `key` is value from from `yb_password_key` and password is value from `yb_password`.

By default this value is not set.

3.1.71 yb_path_astar_post_smooth

Enables post-smoothing for A*. Reduces zig-zags on paths at cost of some CPU cycles.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.72 yb_path_danger_factor_min

Lower bound of danger factor that used to add additional danger to path based on practice.

The `yb_path_danger_factor_min` and `yb_path_danger_factor_max` cvars are used to make dangerous paths even more dangerous based on a random value between these cvars.

It means that bots will be less likely to build paths through these nodes.

These cvars do not affect the goal node, because despite the danger, the bots will go to this node in order to complete the goal (plant a bomb, take a hostage, etc.)

Minimum value is `100`, Maximum value is `2400`, Default value is `200`.

3.1.73 yb_path_danger_factor_max

Upper bound of danger factor that used to add additional danger to path based on practice.

Minimum value is `200`, Maximum value is `4800`, Default value is `400`.

3.1.74 yb_path_dijkstra_simple_distance

Use simple distance path calculation instead of running full Dijkstra path cycle. Used only when Floyd matrices unavailable due to memory limit.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.75 yb_path_floyd_memory_limit

Limit maximum floyd-warshall memory (megabytes). Use Dijkstra if memory exceeds.

Minimum value is `0`, Maximum value is `32`, Default value is `6`.

3.1.76 yb_path_heuristic_type

Selects the heuristic function mode. For debug purposes only.

Minimum value is `0`, Maximum value is `4`, Default value is `0`.

3.1.77 yb_path_randomize_on_round_start

Randomize pathfinding on each round start.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.78 yb_pickup_ammo_and_kits

Allows bots pickup mod items like ammo, health kits and suits.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.79 yb_pickup_best

Allows or disallows bots to pickup best weapons. (Disabling can be useful for some mods with non-pickable weapons).

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.80 yb_pickup_custom_items

Allows or disallows bots to pickup custom items.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.81 yb_ping_base_min

Lower bound for base bot ping shown in scoreboard. Affects only on newly created bots.

Minimum value is `0`, Maximum value is `100`, Default value is `7`.

3.1.82 yb_ping_base_max

Upper bound for base bot ping shown in scoreboard. Affects only on newly created bots.

Minimum value is `0`, Maximum value is `100`, Default value is `34`.

3.1.83 yb_quota

Determines the total number of bots in the game.

Minimum value is `0`, Maximum value is `32`, Default value is `9`.

3.1.84 yb_quota_adding_interval

Interval in which bots are added to the game.

Minimum value is `0.10`, Maximum value is `1`, Default value is `0.10`.

3.1.85 yb_quota_maintain_interval

Interval on which overall bot quota are checked.

Minimum value is `0.40`, Maximum value is `2`, Default value is `0.40`.

3.1.86 yb_quota_mode

Determines the type of how `yb_quota` works.

- If set `fill`, the server will adjust bots to keep N players in the game, where N is `yb_quota`.
- If set `match`, the server will maintain a 1:N ratio of humans to bots, where N is `yb_quota`.
- If set `normal`, this variable does not affect `yb_quota`.

Allowed values is `normal`, `fill` and `match`, Default value is `normal`.

3.1.87 yb_quota_match

Determines the total number of bots in the game, when the `yb_quota_mode` is set to `match`, i.e. for every human, N bots join.

Minimum value is `0`, Maximum value is `32`, Default value is `0`.

3.1.88 yb_radio_mode

Specifies the way bots are talking to each other and player.

- If set to `0` bots will not communicate at all.
- If set to `1` bots will use only the radio.
- If set to `2` bots will use chatter.

Note: Chatter will be used only if bot will find valid wave files in specified in `yb_chatter_path` directory.

Minimum value is `0`, Maximum value is `2`, Default value is `2`.

3.1.89 yb_random_knife_attacks

Allows or disallows the ability for random knife attacks when bot is rushing and no enemy is nearby.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.90 yb_restricted_weapons

A list of individual weapons that are restricted for bot to buy. Separated by semicolon.

The list of weapons for Counter-Strike 1.6:

```
usp - HK USP .45 Tactical
glock - Glock18 Select Fire
deagle - Desert Eagle .50AE
p228 - SIG P228
elite - Dual Beretta 96G Elite
fn57 - FN Five-Seven
m3 - Benelli M3 Super90
xm1014 - Benelli XM1014
mp5 - HK MP5-Navy
tmp - Steyr Tactical Machine Pistol
p90 - FN P90
mac10 - Ingram MAC-10
ump45 - HK UMP45
ak47 - Automat Kalashnikov AK-47
galil - IMI Galil
famas - GIAT FAMAS
sg552 - Sig SG-552 Commando
m4a1 - Colt M4A1 Carbine
aug - Steyr Aug
scout - Steyr Scout
awp - AI Arctic Warfare/Magnum
g3sg1 - HK G3/SG-1 Sniper Rifle
sg550 - Sig SG-550 Sniper
m249 - FN M249 Para
flash - Concussion Grenade
hegren - High-Explosive Grenade
sgren - Smoke Grenade
vest - Kevlar Vest
vesthelm - Kevlar Vest and Helmet
defuser - Defuser Kit
shield - Tactical Shield
```

By default this value is not set.

3.1.91 yb_rotate_bots

Randomly disconnect and connect bots, simulating players join/quit.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.92 yb_rotate_stay_max

Specifies maximum amount of seconds bot keep connected, if rotation active.

Minimum value is `1800`, Maximum value is `14400`, Default value is `3600`.

3.1.93 yb_rotate_stay_min

Specifies minimum amount of seconds bot keep connected, if rotation active.

Minimum value is `120`, Maximum value is `7200`, Default value is `360`.

3.1.94 yb_save_bots_names

Allows to save bot names upon changelevel, so bot names will be the same after a map change

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.95 yb_shoots_thru_walls

Determines the method how bots checks if wall/obstacle is penetrable.

If set to `1` bots will try to shoot through walls more actively, even unrealistically. If set to `2` bots will use algorithm from original PODBot, and shoot through walls less.

The `2` method is consuming a bit more CPU power than the `1` method.

Minimum value is `1`, Maximum value is `3`, Default value is `2`.

3.1.96 yb_show_avatars

Enables or disables displaying bot avatars in front of their names in scoreboard. Note, that is currently you can see only avatars of your steam friends.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.97 yb_show_latency

Determines the type of bots ping displayed.

- If set to `0` there will be no anything in scoreboard about bot ping.
- If set to `1` there will be "BOT" displayed for every bot in scoreboard.
- If set to `2` there will be "fake" ping displayed for every bot in scoreboard.

Minimum value is `0`, Maximum value is `2`, Default value is `2`.

3.1.98 yb_smoke_grenade_checks

Determines the method of how smoke affects the bot's vision

- If set to `0` the smoke won't affect the bot's vision.
- If set to `1` the smoke affection method from PODBot will be used.
- If set to `2` the smoke affection method from official CSBot (a.k.a. ZBot) will be used.

ZBot's method allow the bot to shoot at enemies in smoke clouds under certain circumstances, while PODBot's certainly cannot.

Minimum value is `0`, Maximum value is `2`, Default value is `2`.

3.1.99 yb_spraypaints

If not zero, bots will spray some paints all over the map.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.100 yb_stab_close_enemies

If not zero, bots will stab the enemy with knife if bot is in good condition.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.101 yb_think_fps

Determines how many times per second the rest of bot AI is executed. Higher values will give more smooth movement, but will cause CPU waste, and may cause problems with dedicated servers that have more than 500 fps.

Minimum value is `24`, Maximum value is `90`, Default value is `26`.

3.1.102 yb_think_fps_disable

Allows to completely disable think fps on Xash3D.

It avoids laggy movement of bots when watching them in spectator mode.

Note: This cvar goes into effect when new bots are added.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.103 yb_threadpool_workers

Maximum number of threads bot will run to process some tasks. -1 means half of CPU cores used.

Minimum value is `-1`, Maximum value is `count of your CPU's threads`, Default value is `-1`.

3.1.104 yb_tkpunish

If not zero, bots will punish teammates that attacks the bot.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.105 yb_use_engine_pvs_check

Use engine to check potential visibility of an enemy.

It reduces the number of calls to engine functions, thus reducing CPU usage. The side effect is that bots can fire through obstacles more blatantly.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.1.106 yb_user_follow_percent

Determines percentage of bots that will try to automatically follow the leader. Bots treats bomb guy, vip and human players as leader.

Minimum value is `0`, Maximum value is `100`, Default value is `20`.

3.1.107 yb_user_max_followers

Determines how many bots can respond to human player on `Follow Me` command and follow the human.

Minimum value is `0`, Maximum value is `16`, Default value is `1`.

3.1.108 yb_walking_allowed

If not zero, bots will use “shift” or walking when hearing the nearby enemy.

Minimum value is `0`, Maximum value is `1`, Default value is `1`.

3.1.109 yb_whose_your_daddy

Enables or disables extra hard difficulty for bots.

It zeroes out any reaction or surprise timers, and allows bots to ignore FOV when searching for enemies. It also forces the bots to aim their weapon at the enemy almost immediately.

Minimum value is `0`, Maximum value is `1`, Default value is `0`.

3.2 Map Specific Configuration

Map specific configs where user-configured cvars are stored. The filename is `mapname.cfg` where "mapname" is name of the map for which this config was created eg: `de_dust.cfg` for de_dust map. This file located at `addons/yapb/conf/maps` directory.

You can use the cvars shown above to write in this config. They will be executed automatically when you start a map specified in the name of this config.

By default YaPB does not have any configs for maps.

3.3 Custom Configuration File

A custom config file that allows you to change some hard-coded things in the bot code.

It's located in `addons/yapb/conf/custom.cfg`

`C4ModelName` - It sets a custom name for C4 model, for servers that replace C4 model with it's own. By default it's `c4.mdl` (the models/path is omitted), so if you need to use `models/mybomb/mybomb.mdl`, you should specify `mybomb/mybomb.mdl`.

`AMXParachuteCvar` - It sets a custom cvar name for parachute handling, there are various plugins that handles parachute (AMX Parachute, AMX Parachute Lite, etc.), you can specify needed cvar here.

`CustomCSDMSpawnPoint` - It sets a custom spawn point classname for CSDM mods that add custom spawn points other than info_player_start and info_player_deathmatch. So bots will be able to join the game without default spawn entities.

3.4 Bot Difficulty Configuration

You can fine-tune the bots thanks to the difficulty configuration file. It contains the values of reaction time, headshot and wallshot probabilities, recoil control values, and aim offset axes.

Each of these values is tied to each difficulty level.

The bots difficulty configuration file is located on the path `addons/yapb/conf/difficulty.cfg`

The valid format is:

```
Level = minReactionTime(s),maxReactionTime(s),headshotProbability,seenThruWallChance,
↪heardThruWallChance,maxWeaponRecoil,aimError
```

Where:

```
minReactionTime - Minimal time in seconds from when the bot first saw the enemy to when␣
↪it can recognize it.
maxReactionTime - Same as above, but upper cap of the limit.
headshotProbability - The probability that the bot will aim at the head instead of body,␣
↪if both body and head are visible.
seenThruWallChance - Chance that the bot will attack the enemy if it believes that it is␣
↪there and just saw it.
heardThruWallChance - Chance that the bot will attack the enemy if it believes that it␣
↪is there and just heard it.
```

(continues on next page)

(continued from previous page)

```
maxWeaponRecoil - Maximum weapon recoil to compensate by pausing fire.
aimError - (x, y, z) offsets to add aim error to bot aiming
```

Example:

```
Expert = 0.1, 0.2, 100, 90, 90, 21, 0.0, 0.0, 0.0
```

3.5 Logos Configuration

List of the bot spray paints are stored is in `addons/yapb/conf/logos.cfg`. It uses textures from decals.wad, so if you want to add a new spray paint texture, it should be in that file. To see available textures in decals.wad you can use special tools like Wally, Half-Life Texture Tools, etc.

Default spray paints list:

```
{biohaz
{graf003
{graf004
{graf005
{lambda06
{target
{hand1
{spit2
{bloodhand6
{foot_l
{foot_r
```

3.6 Weapons Configuration

Main config where most stuff regarding weapon handling is defined. The filename is `weapon.cfg` and located at `addons/yapb/conf` directory.

To edit this file you need to know the weapon numbering.

3.6.1 MapStandard Field

This field is by default used on all maps except VIP scenario.

Below is the table that specifies which Team is allowed to buy a weapon on a Map. You can also use it to allow/disallow Weapons for a Team or a Map/Gamemode (remember that some Weapons are team-specific and can't be bought by another team).

Weapon buy flags:

```
-1 = Disallow Buying for any Team
0 = Terrorist Team only
1 = CT Team only
2 = Can be bought by both Teams
```

Example:

```
MapStandard = -1,0,-1,2,-1,0,1,2,2,2,-1,2,-1,-1,0,0,1,0,1,1,2,2,0,1,2,1
```

3.6.2 MapAS Field

This field is by default used only on VIP scenario maps.

Below is the table that specifies which Team is allowed to buy a weapon on a Map. You can also use it to allow/disallow Weapons for a Team or a Map/Gamemode (remember that some Weapons are team-specific and can't be bought by another team).

Weapon buy flags:

```
-1 = Disallow Buying for any Team
0 = Terrorist Team only
1 = CT Team only
2 = Can be bought by both Teams
```

Example:

```
MapAS = -1,-1,-1,2,-1,0,1,1,1,1,1,1,0,2,0,-1,1,0,1,1,0,0,-1,1,1,1
```

3.6.3 Grenade buying percentage

Specifies the buying percents for grenade inventory.

From left to right:

```
1 - HE grenade.
2 - Flashbang.
3 - Smoke grenade.
```

Example:

```
GrenadePercent = 98,75,60
```

3.6.4 Bot Economics

Specifies economics values for buying the weapons.

From left to right:

```
1 - If bot has more money than the value specified here, it can buy the primary weapon.
2 - If bot has more money than the value specified here, it will not buy SMGs (MP5,␣
↪MAC10, TMP, P90, UMP45, SCOUT) (only for CTs) (+ 8/9/10)
3 - If bot has more money than the value specified here, it will not buy SMGs (MP5,␣
↪MAC10, TMP, P90, UMP45, SCOUT) (only for Ts) (+ 8/9/10))
4 - If bot has more money than the value specified here, it can buy shotguns (M3,␣
↪XM1014).
5 - If bot has less money than the value specified here, it cannot buy shotguns (M3,␣
↪XM1014).
6 - If bot has more money than the value specified here, it can buy AWP, SG550, G3SG1,␣
```

(continues on next page)

(continued from previous page)

```
↪M249.
7 - If bot has less money than the value specified here, it cannot buy AWP, SG550, G3SG1,
↪ M249.
8 - How much money bot leaves in stock, at purchase of the primary weapon (only for type␣
↪of behaviour - Normal)
9 - How much money bot leaves in stock, at purchase of the primary weapon (only for type␣
↪of behaviour - Rusher).
10 - How much money bot leaves in stock, at purchase of the primary weapon  (only for␣
↪type of behaviour - Careful).
11 - If bot has more money than the value specified here, it can buy the shield.
```

Example:

```
Economics = 1550,2100,2100,4000,6000,7000,16000,1200,800,1100,3000
```

3.6.5 Weapon Priorities

This table stores the Weapon Priorities of the Bots depending on Personality (it affects buying & picking up better weapons from the ground).

Numbering of the weapons:

```
0 - KNIFE
1 - USP
2 - GLOCK18
3 - DEAGLE
4 - P228
5 - ELITE
6 - FIVESEVEN
7 - M3
8 - XM1014
9 - MP5NAVY
10 - TMP
11 - P90
12 - MAC10
13 - UMP45
14 - AK47
15 - SG552
16 - M4A1
17 - GALIL
18 - FAMAS
19 - AUG
20 - SCOUT
21 - AWP
22 - G3SG1
23 - SG550
24 - M249
25 - SHIELD
```

From left to right. Rightmost value is a most wanted bot weapon. Leftmost value is most worst weapon.

Note: Knife should most "worst" weapon in this table, otherwise things are messed up.

Examples:

```
PersonalityNormal = 00,02,01,04,05,06,03,12,10,24,25,13,11,08,07,22,23,20,21,09,19,15,17,
↪18,14,16
PersonalityRusher = 00,02,04,05,01,06,03,24,25,22,23,20,10,12,13,07,08,21,11,09,15,19,17,
↪18,16,14
PersonalityCareful = 00,02,01,04,05,06,03,07,08,12,10,13,11,09,18,17,15,19,16,14,20,22,
↪25,23,24,21
```

CHAPTER

FOUR

CUSTOMIZATION

4.1 Chat customization

You can create a new chat base for your language or change the existing one. It is located in the folder: `addons/yapb/conf/lang` To do this, create a file `**_chat.cfg` or open an existing one.

Bots can write chat messages depending on the situation: planting a bomb, killing an enemy, attacking teammates, etc.

4.1.1 Placeholders

- `%v` - inserts the nickname of the last killed player
- `%f` - inserts the nickname of the team leader
- `%t` - inserts the nickname of the alive teammate
- `%e` - inserts the nickname of the alive enemy
- `%s` - inserts the nickname of the random player
- `%m` - inserts the name of the current map
- `%d` - inserts the mod name from `liblist.gam` file (Counter-Strike or Condition Zero)

4.1.2 Chat sections

`[BOMBPLANT]` - Sets a list of messages that bots will write after planting a bomb.

`[KILLED]` - Sets a list of messages for bots that will write after killing an enemy. Use the placeholder `%v` to write the nickname of the killed enemy. Example:

```
[KILLED]
You're dead %v!
```

When the bot kills the enemy with the nickname "John Smith", he will write "You're dead John Smith!" using the line shown in the example.

`[WELCOME]` - Sets the list of messages that the bot will write when it connects to server.

`[TEAMATTACK]` - Sets the list of messages that the bot will write when attacked by a teammate.

`[TEAMKILL]` - Sets the list of messages that the bot will write when it killed a teammate.

`[DEADCHAT]` - Sets the list of messages that the bot will write when it is dead or is in spectator mode.

Note: Minimum number is 9! If you write less than 9 messages for this trigger, yapb will crash!

`[REPLIES]` - Sets the list of messages that the bot will write in answer to another bot if it has an answer to the specified word To set a word to which the answer will be, you need to set a key to specified words separated by commas Example:

```
[REPLIES]
@KEY "WORD", "ANOTHER WORD"
This is the answer to the given words.
This is another answer to the given words.

@KEY "KEYWORD"
This is the answer to new key word.
```

This is how it will look in the game:

```
John Smith: Bla bla bla word
Ricardo Milos: This is answer to the given words.

Keanu Reeves: Bla bla bla another word bla bla...
Tommy Vercetti: This is another answer to the given words.

Ryan Gosling: Answer me a new keyword.
Soap MacTavish: This is the answer to new key word.
```

Bots can use these replies at random.

Warning: Please note that the key words in the [REPLIES] trigger must be written in capital letters! In messages, they can be written in any format.

`[UNKNOWN]` - Sets the list of messages that the bot will write in answer to another bot if it has no response in the [REPLIES] trigger

4.2 VoiceChat customization

YaPB supports voice chat as well as zBot. All paths for yapb voice chat audio files are in the file: `chatter.cfg` which located in the folder `addons/yapb/conf`.

`RewritePath` sets the folder where the voice chat audio files are located. By default it is `sound/radio/bot`

4.2.1 Radio events

`Event Radio_***` sets the name of the sound files that bot will speak instead of using a specific radio commands.

Which of them:

- `Event Radio_CoverMe` - “Cover Me!” radio command.
- `Event Radio_YouTakePoint` - “You Take the Point.” radio command.
- `Event Radio_HoldPosition` - “Hold This Position.” radio command.
- `Event Radio_RegroupTeam` - “Regroup Team.” radio command.

- `Event Radio_FollowMe` - "Follow Me." radio command.
- `Event Radio_TakingFire` - "Taking Fire…Need Assistance!" radio command.
- `Event Radio_GoGoGo` - "Go go go!" radio command.
- `Event Radio_Fallback` - "Team, fall back!" radio command.
- `Event Radio_StickTogether` - "Stick together, team." radio command.
- `Event Radio_GetInPosition` - "Get in position and wait for my go." radio command.
- `Event Radio_StormTheFront` - "Storm the Front!" radio command.
- `Event Radio_ReportTeam` - "Report in, team." radio command.
- `Event Radio_Affirmative` - "Affirmative./Roger that." radio command.
- `Event Radio_EnemySpotted` - "Enemy spotted." radio command.
- `Event Radio_NeedBackup` - "Need backup." radio command.
- `Event Radio_SectorClear` - "Sector clear." radio command.
- `Event Radio_InPosition` - "I'm in position." radio command.
- `Event Radio_ReportingIn` - "Reporting in." radio command.
- `Event Radio_ShesGonnaBlow` - "Get out of there, it's gonna blow!" radio command.
- `Event Radio_Negative` - "Negative." radio command.
- `Event Radio_EnemyDown` - "Enemy down." radio command.

You can comment out these lines if you want the bot to use standard radio commands.

4.2.2 Chatter events

`Event Chatter_***` sets the names of sound files for bot's chatter that it will speak.

Chatter events list:

- `Event Chatter_DiePain` - bot death sounds.
- `Event Chatter_GoingToPlantBomb` - bot says it's going to plant a bomb.
- `Event Chatter_GoingToGuardVIPSafety` - bot says that he is going to guard the vip escape zone.
- `Event Chatter_RescuingHostages` - bot says that he is rescuing hostages.
- `Event Chatter_TeamKill` - bot reaction to killing a teammate.
- `Event Chatter_GuardingVipSafety` - bot says that he is guarding the vip escape zone.
- `Event Chatter_PlantingC4` - bot says it's planting a bomb.
- `Event Chatter_InCombat` - bot says that he is fighting with the enemy right now.
- `Event Chatter_SeeksEnemy`- bot says that he is waiting for the enemy.
- `Event Chatter_Nothing` - bot says that there is no one in this sector.
- `Event Chatter_EnemyDown` - bot says that he killed the enemy.
- `Event Chatter_UseHostage` - bot says that he took a hostage.
- `Event Chatter_WonTheRound` - bot's reaction to win.
- `Event Chatter_QuicklyWonTheRound` - bot's reaction to a quick win.

- Event Chatter_NoEnemiesLeft - bot says that there are no more remaining enemies.
- Event Chatter_FoundBombPlace - bot says that he found a place with a planted bomb.
- Event Chatter_WhereIsTheBomb - bot asks where the bomb is.
- Event Chatter_DefendingBombSite - bot says it's defending the bomb site.
- Event Chatter_BarelyDefused - bot's reaction to a barely defused bomb.
- Event Chatter_NiceshotCommander - bot's reaction to a nice shot by a player.
- Event Chatter_ReportingIn - bot says it's reporting in.
- Event Chatter_SpotTheBomber - bot says that he noticed a bomber.
- Event Chatter_VIPSpotted - bot says that he noticed the VIP.
- Event Chatter_FriendlyFire - bot reaction when attacked by a teammate.
- Event Chatter_GotBlinded - bot reaction to flashbang.
- Event Chatter_GuardDroppedC4 - bot says that he guards the dropped C4.
- Event Chatter_DefusingC4 - bot says that he is defusing C4.
- Event Chatter_FoundC4 - bot says that he is found C4.
- Event Chatter_ScaredEmotion - bot reaction when he met several enemies and there are no teammates nearby.
- Event Chatter_HeardEnemy - bot says that he heard the enemy.
- Event Chatter_SniperWarning - bot warns about sniper.
- Event Chatter_SniperKilled - bot reports that he killed a sniper.
- Event Chatter_OneEnemyLeft - bot says that there is only one enemy left.
- Event Chatter_TwoEnemiesLeft - bot says that there are two enemies left.
- Event Chatter_ThreeEnemiesLeft - bot says that there are three enemies left.
- Event Chatter_NiceshotPall - bot's reaction to a nice shot from another bot.
- Event Chatter_GoingToGuardHostages - bot says that he is going to guard the hostages.
- Event Chatter_GoingToGuardDroppedBomb - bot says that he is going to guard the dropped bomb.
- Event Chatter_OnMyWay - bot says it will be here soon.
- Event Chatter_LeadOnSir - bot tells the player that he will follow him.
- Event Chatter_Pinned_Down - bot asks for help from teammates when they are nearby.
- Event Chatter_GottaFindTheBomb - bot says he found a bomb.
- Event Chatter_You_Heard_The_Man - bot talks about the beginning of the round (currently not used).
- Event Chatter_Lost_The_Commander - bot says that the commander (player) was killed, concerns the career mode in Counter-Strike Condition Zero.
- Event Chatter_NewRound - same as Chatter_You_Heard_The_Man (currently not used).
- Event Chatter_CoverMe - bot asks to be covered.
- Event Chatter_BehindSmoke - bot says that he is behind the smoke (currently not used).
- Event Chatter_BombSiteSecured - bot says that he defused the bomb.
- Event Chatter_GoingToCamp - bot says that he is going to camp (guard an area).

- `Event Chatter_Camp` - bot says he's camping.

How this file should look like:

```
RewritePath sound/radio/bot

Event Radio_CoverMe = ("cover_me", "cover_me2");
// Event Radio_YouTakePoint = ("");
// Event Radio_HoldPosition = ("");
// Event Radio_RegroupTeam = ("");
Event Radio_FollowMe = ("lead_on_sir", "lead_the_way_sir", "lead_the_way", "ok_sir_lets_
↪go", "lead_on_commander", "lead_the_way_commander", "ok_cmdr_lets_go");
Event Radio_TakingFire = ("taking_fire_need_assistance2", "i_could_use_some_help", "i_
↪could_use_some_help_over_here", "help", "need_help", "need_help2", "im_in_trouble");

// Event Radio_GoGoGo = ("");
// Event Radio_Fallback = ("");
// Event Radio_StickTogether = ("");
// Event Radio_GetInPosition = ("");
// Event Radio_StormTheFront = ("");
Event Radio_ReportTeam = ("report_in_team", "anyone_see_them", "anyone_see_anything",
↪"where_are_they", "where_could_they_be");

Event Radio_Affirmative = ("affirmative", "no2", "roger_that", "me_too", "ill_come_with_
↪you", "ill_go_with_you", "ill_go_too", "i_got_your_back", "i_got_your_back2", "im_with_
↪you", "im_with_you", "sounds_like_a_plan", "good_idea");
Event Radio_EnemySpotted = ("one_guy", "two_of_them", "theyre_all_over_the_place2", "the_
↪actions_hot_here", "its_a_party");
Event Radio_NeedBackup = ("taking_fire_need_assistance2", "i_could_use_some_help", "i_
↪could_use_some_help_over_here", "help", "need_help", "need_help2", "im_in_trouble");
Event Radio_SectorClear = ("clear", "clear2", "clear3", "clear4", "area_clear", "all_
↪clear_here", "nothing_happening_over_here", "nothing_here", "theres_nobody_home");
Event Radio_InPosition = ("lets_wait_here", "lets_hold_up_here_for_a_minute", "im_gonna_
↪hang_back", "im_going_to_wait_here", "im_waiting_here");
Event Radio_ReportingIn = ("reporting_in");
// Event Radio_ShesGonnaBlow = ("");
Event Radio_Negative = ("ahh_negative", "negative", "no2", "negative2", "i_dont_think_so
↪", "naa", "no_thanks", "no", "nnno_sir", "no_sir");
Event Radio_EnemyDown = ("enemy_down", "enemy_down2");

// end of radio, begin some voices (NOT SORTED)
Event Chatter_SpotTheBomber = ("i_see_the_bomber", "theres_the_bomber", "hes_got_the_bomb
↪", "hes_got_the_bomb2", "hes_got_the_package", "spotted_the_delivery_boy");
Event Chatter_FriendlyFire = ("cut_it_out", "what_are_you_doing", "stop_it", "ow_its_me",
↪ "ow", "ouch", "im_on_your_side", "hold_your_fire", "hey", "hey2", "ouch", "ouch",
↪"ouch");
Event Chatter_DiePain = ("pain2", "pain4", "pain5", "pain8", "pain9", "pain10");
Event Chatter_GotBlinded = ("ive_been_blinded", "my_eyes", "i_cant_see", "im_blind");
Event Chatter_GoingToPlantBomb = ("im_gonna_go_plant", "im_gonna_go_plant_the_bomb");
Event Chatter_RescuingHostages = ("the_hostages_are_with_me", "taking_the_hostages_to_
↪safety", "ive_got_the_hostages", "i_have_the_hostages");
Event Chatter_GoingToCamp = ("im_going_to_camp");
Event Chatter_HearSomething = ("hang_on_i_heard_something", "i_hear_something", "i_heard_
↪them", "i_heard_something_over_there");
```

(continues on next page)

(continued from previous page)

```
Event Chatter_TeamKill = ("what_happened", "noo", "oh_my_god", "oh_man", "oh_no_sad",
↪"what_have_you_done");
Event Chatter_ReportingIn = ("reporting_in");
Event Chatter_GuardDroppedC4 = ("bombsite", "bombsite2", "i_got_a_covered", "im_camping_c
↪");
Event Chatter_Camp = ("im_waiting_here");
Event Chatter_PlantingC4 = ("planting_the_bomb", "planting");
Event Chatter_DefusingC4 = ("defusing", "defusing_bomb", "defusing_bomb");
Event Chatter_InCombat = ("attacking", "attacking_enemies", "engaging_enemies", "in_
↪combat", "in_combat2", "returning_fire");
Event Chatter_SeeksEnemy = ("lets_wait_here", "lets_hold_up_here_for_a_minute", "im_
↪gonna_hang_back", "im_going_to_wait_here", "im_waiting_here");
Event Chatter_Nothing = ("nothing_here", "nothing");
Event Chatter_EnemyDown = ("hes_dead", "hes_down", "got_him", "dropped_him", "killed_him
↪", "ruined_his_day", "wasted_him", "made_him_cry", "took_him_down", "took_him_out2",
↪"took_him_out", "hes_broken", "hes_done");
Event Chatter_UseHostage = ("talking_to_hostages", "rescuing_hostages");
Event Chatter_FoundC4 = ("bombs_on_the_ground", "bombs_on_the_ground_here", "the_bomb_is_
↪down", "the_bomb_is_on_the_ground", "they_dropped_the_bomb");
Event Chatter_WonTheRound = ("good_job_team", "nice_work_team", "way_to_be_team", "well_
↪done");
Event Chatter_QuicklyWonTheRound = ("i_am_dangerous", "do_not_mess_with_me", "we_owned_
↪them", "they_never_knew_what_hit_them", "thats_the_way_this_is_done", "and_thats_how_
↪its_done", "owned", "yesss", "yesss2", "yea_baby", "whoo", "whoo2", "oh_yea");
Event Chatter_ScaredEmotion = ("whoa", "uh_oh", "oh_no", "yikes", "oh", "oh_boy", "oh_
↪boy2", "aah");
Event Chatter_HeardEnemy = ("i_hear_them", "hang_on_i_heard_something", "i_hear_something
↪", "i_heard_them", "i_heard_something_over_there");
Event Chatter_SniperWarning = ("sniper", "sniper2", "watch_it_theres_a_sniper");
Event Chatter_SniperKilled = ("got_the_sniper", "got_the_sniper2", "sniper_down", "took_
↪out_the_sniper", "the_sniper_is_dead");
Event Chatter_VIPSpotted = ("i_see_our_target", "target_spotted", "target_acquired");
Event Chatter_GuardingVipSafety = ("watching_the_escape_route", "im_at_the_escape_zone",
↪"watching_the_escape_zone", "guarding_the_escape_zone", "guarding_the_escape_zone2");
Event Chatter_GoingToGuardVIPSafety = ("im_going_to_cover_the_escape_zone", "im_going_to_
↪watch_the_escape_zone", "im_going_to_keep_an_eye_on_the_escape", "heading_to_the_
↪escape_zone");
Event Chatter_OneEnemyLeft = ("one_guy_left", "theres_one_left");
Event Chatter_TwoEnemiesLeft = ("two_enemies_left", "two_to_go");
Event Chatter_ThreeEnemiesLeft = ("three_left", "three_to_go", "three_to_go2");
Event Chatter_NoEnemiesLeft = ("that_was_the_last_one", "that_was_it", "that_was_the_
↪last_guy");
Event Chatter_FoundBombPlace = ("theres_the_bomb", "theres_the_bomb2");
Event Chatter_WhereIsTheBomb = ("wheres_the_bomb", "wheres_the_bomb2", "wheres_the_bomb3
↪", "where_is_it");
Event Chatter_DefendingBombSite = ("bombsite", "bombsite2", "im_camping_b", "heading_to_c
↪");
Event Chatter_BarelyDefused = ("i_wasnt_worried_for_a_minute", "that_was_a_close_one",
↪"well_done", "whew_that_was_close");
Event Chatter_NiceshotCommander = ("good_one_sir", "good_one_sir2", "nice_shot_sir",
↪"nice_one_sir");
Event Chatter_NiceshotPall = ("good_one", "good_one2", "nice_shot", "nice_shot2", "good_
```

(continues on next page)

(continued from previous page)

```
↪shot", "good_shot2", "nice", "nice2", "very_nice");
Event Chatter_GoingToGuardHostages = ("camping_hostages", "im_going_to_camp_the_hostages
↪", "im_going_to_guard_the_hostages", "im_going_to_guard_the_hostages2");
Event Chatter_GoingToGuardDoppedBomb = ("im_going_to_guard_the_bomb", "im_going_to_guard_
↪the_bomb2", "im_going_to_keep_an_eye_on_the_bomb", "im_going_to_watch_the_bomb");
Event Chatter_OnMyWay = ("on_my_way", "on_my_way2", "im_coming", "hang_on_im_coming",
↪"be_right_there");
Event Chatter_LeadOnSir = ("lead_on_sir", "lead_the_way_sir", "lead_the_way", "ok_sir_
↪lets_go", "lead_on_commander", "lead_the_way_commander", "ok_cmdr_lets_go");
Event Chatter_Pinned_Down = ("they_got_me_pinned_down_here", "im_pinned_down");
Event Chatter_GottaFindTheBomb = ("theres_the_bomb", "theres_the_bomb2");
Event Chatter_Lost_The_Commander = ("weve_lost_the_commander", "the_commander_is_down",
↪"the_commander_is_down_repeat");
Event Chatter_CoverMe = ("cover_me", "cover_me2");
Event Chatter_BombSiteSecured = ("i_wasnt_worried_for_a_minute", "that_was_a_close_one",
↪"well_done", "whew_that_was_close");
```

CHAPTER

FIVE

BOT USAGE

5.1 The YaPB User Menu (yb menu)

5.1.1 Main Menu

Pressing the "=" key in game a menu with the following options should appear on your screen

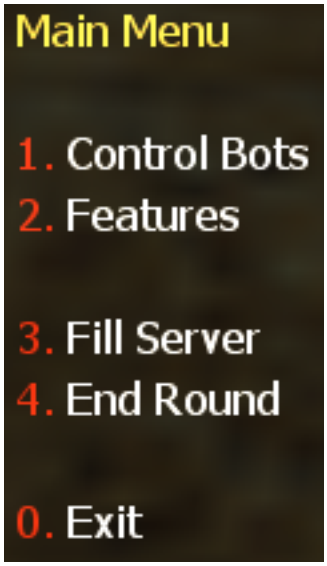

Fig. 1: This is the YaPB user menu

1. **Control Bots** - A menu that adds or removes bots from the game.
2. **Features** - A menu that configures the type of weapons used by bots, opens the waypoints menu, toggles debug mode and controls bots commands.
3. **Fill Server** - A menu that fills the server with bots with the specified parameters.
4. **End Round** - Kills all bots for the end of round.

5.1.2 Bots Control Menu

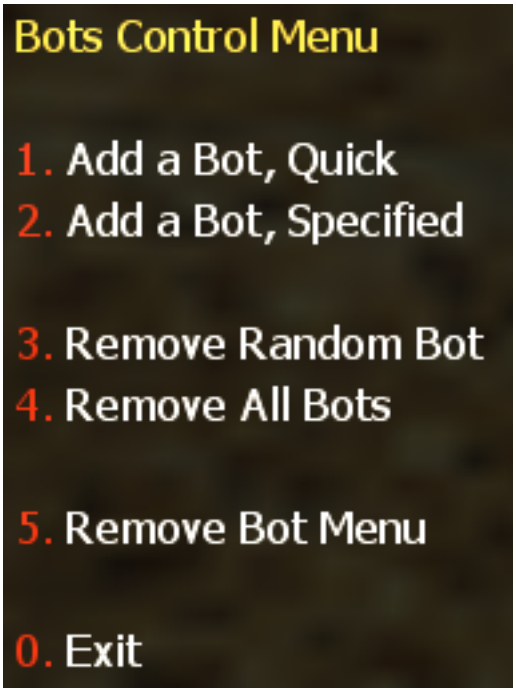

Fig. 2: Bots Control Menu

1. **Add a Bot, Quick** - This does what it says. It quickly adds a Bot giving him a random name, team, difficulty and model. Difficulty will be chosen randomly between your yb_difficulty_min/yb_difficulty_max values specified in yapb.cfg.
2. **Add a Bot, Specified** - Allows you specify all things (except name) for adding a single Bot.
3. **Remove Random Bot** - removes a random bot.
4. **Remove All Bots** - removes all bots from the server.
5. **Remove Bot Menu** - A menu that allows you to remove a bot from the server specified in the list.

5.1.3 Bots Features Menu

1. **Weapon Mode Menu** - A menu that configures the type of weapons used by bots.
2. **Waypoint Menu** - Opens the waypoint menu.
3. **Select Personality** - Adds a bot with the currently set difficulty with personality setting.
4. **Toggle Debug Mode** - Enables or disables debug mode.
5. **Command Menu** - Opens the bot command menu.

1. **Make Double Jump** - Forces the nearest teammate bot to crouch next to you to make double jump.
2. **Finish Double Jump** - Releases the bot after the first command, it must get up and go about his business.
3. **Drop the C4 Bomb** - Forces the bot carrying the bomb to drop it at you.
4. **Drop the Weapon** - Makes the teammate bot throw a weapon at you.

Fig. 3: This lets You choose a difficulty for the specific Bot.

Fig. 4: This lets You choose a personality for the specific Bot.

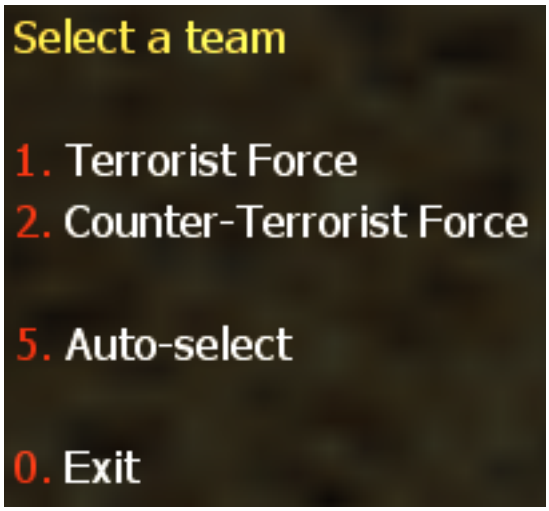

Fig. 5: This lets You choose a team for the specific Bot.

Fig. 6: This lets You choose a class for the specific Bot. (for CT Team)

Fig. 7: This lets You choose a class for the specific Bot. (for T Team)

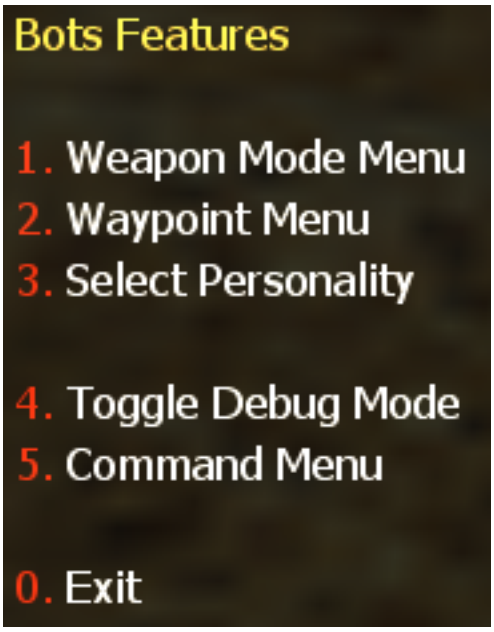

Fig. 8: Bots Features Menu

Fig. 9: Bots Weapon Mode Menu

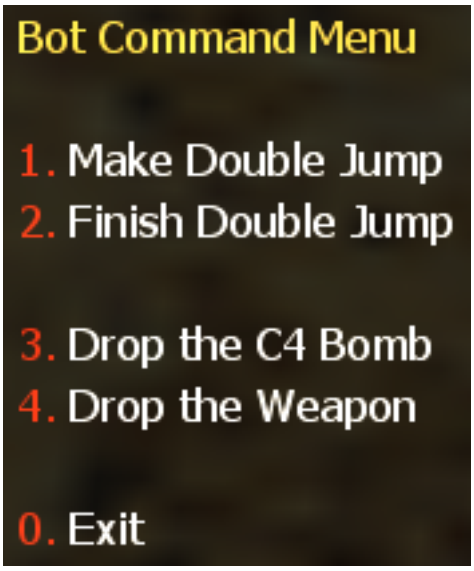

Fig. 10: Bot Command Menu

Note: Bot will only throw a weapon at you when it have a primary weapon and 2000 or more dollars in the account.

5.1.4 Waypoint Menu

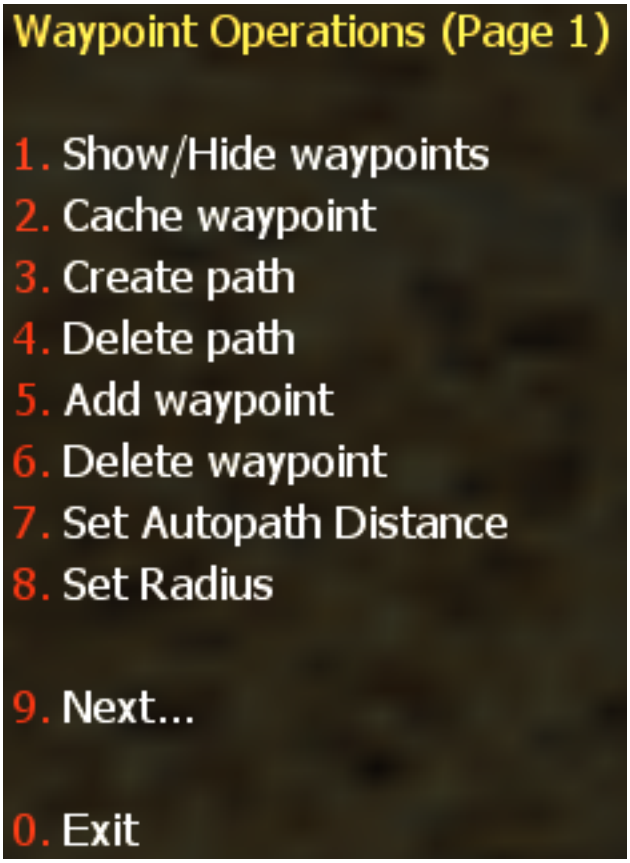

Fig. 11: Waypoint Editor Menu (Page 1)

1. **Show/Hide waypoints** - Show or hide the display of waypoints.
2. **Cache waypoint** - Cache waypoint for future use.
3. **Create path** - Opens the menu for creating path connections.
4. **Delete path** - Removes the path from selected waypoint.
5. **Add waypoint** - Opens the menu for selecting the type for adding waypoint.
6. **Delete waypoint** - Removes the waypoint you are standing on.
7. **Set Autopath Distance** - Opens the autopath distance setting menu.
8. **Set Radius** - Opens the waypoint radius setting menu.
9. **Next...** - Goes to the second page of the waypoint menu.

1. **Debug goal** - Opens a menu for debugging bot walkability to a specified waypoint.
2. **Autowaypoint on/off** - Enables or disables automatic waypoint placement.

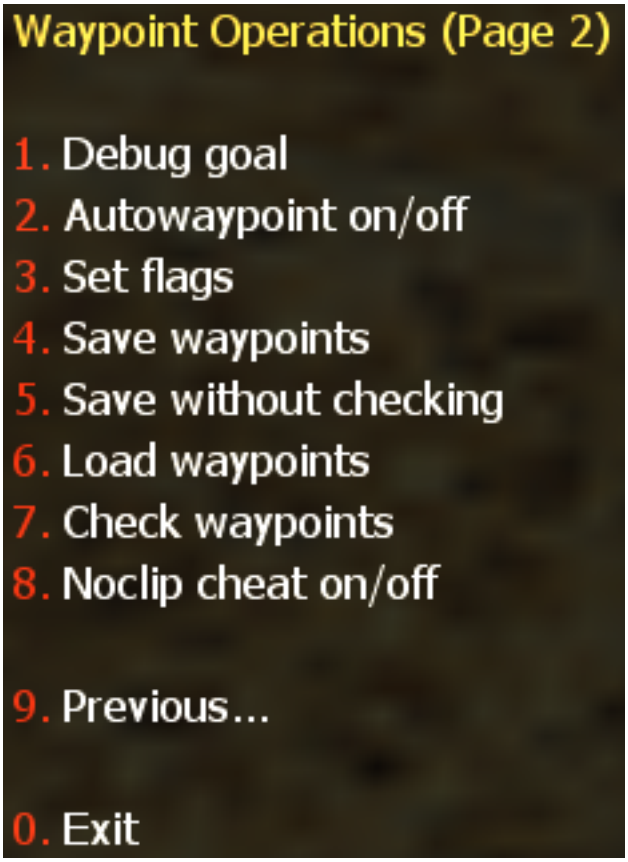

Fig. 12: Waypoint Editor Menu (Page 2)

3. **Set flags** - Opens the menu for selecting flags for a waypoint.
4. **Save waypoints** - Saves waypoints with path integrity check.
5. **Save without checking** - Saves waypoints without checking (because of this, there may be problems with the behavior of bots).
6. **Load waypoints** - Loads waypoints from a file.
7. **Check waypoints** - Checks waypoints for errors.
8. **Noclip cheat on/off** - Enables or disables noclip cheat.
9. **Previous...** - Goes to the first page of the waypoint menu.

5.2 YaPB console commands Summary

The following main YaPB commands are available:

`yb add`	Adds specific bot into the game. (see below)
`yb kick`	Kicks off the random or specified bot from the game. (see below)
`yb removebots`	Kicks all the bots from the game. Also available via alias `yb kickall`. (see below)
`yb kill`	Kills the specified team or all the bots. (see below)
`yb fill`	Fills the server (add bots) with specified parameters. (see below)
`yb vote`	Forces all the bots to vote to specified map.
`yb weapons`	Sets the bots weapon mode to use. (see below)
`yb menu`	Opens the main bot menu.
`yb version`	Displays version information about bot build.
`yb list`	Lists the bots currently playing on server.
`yb cvars`	Displays all the cvars with their descriptions. (see below)
`yb graph`	Handles graph operations.
`yb show_custom`	Shows the current values from custom.cfg
`yb exec`	Executes a client command on bot entity.

To get help for all commands such as arguments, aliases, etc, type in the console `yb help`.

If you want to get help for specified command for example `yb add`, type in the console `yb help add`.

5.2.1 yb add

To add a specific bot to the game, with nickname: John Smith, Difficulty: Average, Personality: Careful, Team: Counter-Terrorists, Team Class: SAS, you should type in console `yb add 1 2 2 3 "John Smith"`

yb add arguments info:

Difficulties

- `0` - Newbie
- `1` - Average
- `2` - Normal
- `3` - Professional
- `4` - Godlike

Personalities

- `0` - Normal
- `1` - Aggressive (rusher)
- `2` - Careful

Teams

- `0` - Random
- `1` - Terrorists
- `2` - Counter-Terrorists

Team classes

Terrorists:

- `0` - Random
- `1` - Phoenix Connexion
- `2` - Elite Crew
- `3` - Arctic Avengers
- `4` - Guerilla Warfare
- `5` - Midwest Militia **(Condition Zero only!)**

Counter-Terrorists:

- `0` - Random
- `1` - Seal Team 6
- `2` - GSG-9
- `3` - SAS
- `4` - GIGN
- `5` - Spetsnaz **(Condition Zero only!)**

Correct format for the `yb add` command is `yb add [difficulty] [personality] [team] [model] [name]`. All bot values are selected by numbers (except the bot name).

5.2.2 yb kick

Type in console `yb kick` command to remove the random bot.

If you want to remove the bot from the specified team, you should type in the console `yb kick t` to kick a bot from Terrorists team, and `yb kick ct` to kick a bot from Counter-Terrorists team.

5.2.3 yb removebots

You can also use the alias `yb kickall` to remove all bots.

If you want to remove bots instantly, add the `instant` argument to this command.

Example: `yb kickall instant`

5.2.4 yb kill

The `yb kill` command kills all the bots. To kill a specific team, such as terrorists you should type in console `yb kill t` command. For Counter-Terrorists the command are `yb kill ct`

The `silent` argument disables the `All bots died...` message in the console. You should enter the `yb kill silent` in console, to kill bots without that message.

5.2.5 yb fill

To fill the server with random bots type in console `yb fill 0`.

If you want to fill the server with specific bots, for example: Team: Terrorists, Count: 5, Difficulty: Normal, Personality: Aggressive, you should type in console the follow command `yb fill 1 5 2 1`.

yb fill arguments info:

Teams

- `0` - Both teams
- `1` - Terrorists only
- `2` - Counter-Terrorists only

Difficulties

- `0` - Newbie
- `1` - Average
- `2` - Normal
- `3` - Professional
- `4` - Godlike

Personalities

- `0` - Normal
- `1` - Aggressive (rusher)
- `2` - Careful

Don't enter the bot personality value if you want bots with random personalities.

Correct format for the `yb fill` command is `yb fill [team] [count] [difficulty] [personality]`.

5.2.6 yb weapons

To force the bot to use only a certain type of weapon, for example, shotguns, you should type in console the `yb weapons shotgun` command.

Allowed values: `knife|pistol|shotgun|smg|rifle|sniper|standard`.

Standard means that bots will use all weapons.

5.2.7 yb cvars

This command lists all cvars with their descriptions.

If you want to save all cvars you configured to config, add the `save` argument to this command.

You can also save a map-specific config by using the `save_map` argument to save the current values of all cvars to `addons/yapb/conf/maps/map_name.cfg`.

Example: `yb cvars save`

Also you can narrow your search by entering a word as an argument, instead of looking through a list of all cvars.

To restore the default values of all bot cvars, type `yb cvars defaults` in console.

5.2.8 yb exec

This command allows you to execute a command from a bot entity.

Valid usage is: `yb exec [user_id] [command]`.

Where `[user_id]` is bot's ID, which you can find by entering the `yb list` command in the console.

5.3 Adding bots to the game

Select `1. Add a Bot, Quick` from the bot control menu to add a bot with random stats (name, difficulty, personality etc.) Select `2. Add a Bot, Specified` from the bot control menu to add a bot with manually specified stats.

Or type in console `yb_quota x` where X is amount of adding bots.

5.4 Selecting the bot language

You must open the file `yapb.cfg` in the folder `addons/yapb/conf` and change the value of yb_language cvar to the next available one.

1. `en` - English Language
2. `ru` - Russian Language
3. `de` - Deutsch Language

For example, write in the config `yb_language ru` for Russian Language.

5.5 Bot management on a dedicated server

To have access to the bot's menus and commands, you need to in a server console specify a password and a key from which the password will be read.

To specify a password, you must enter in the console the following cvar `yb_password botpassword` where `botpassword` is the password you specified. To specify a key, you must enter in the console the following cvar `yb_password_key _ybpw`, where `_ybpw` is the key you specified.

Then, in a client console you must enter the following commandline in the console `setinfo _ybpw botpassword` to have access to the commands and menus of the bot. To have access to graph commands, you need to enter in the console the following command `yb g acquire_editor`. Make sure that no one has entered this command before you, who has the password from the bot. Otherwise, you won't be able to access graph commands until that player removes graph editing rights.

To revoke the rights to edit graphs, you must enter in the console the following command: `yb g release_editor`.

CHAPTER

SIX

WAYPOINTING

6.1 Brief information

6.1.1 Waypoints, what are they?

Unlike humans, bots cannot see a map and analyze what they see. If you see a building with a door, you can walk straight to the door, open it and enter the building. Bots cannot do this without help! They can see and fight enemies or react if they are being attacked. These ways of behaviour work without any external help. But in order to find their way around the map and to safely navigate through all ways and passages, they do need some help. They need something that tells them where they can go and where they can't. They need something that shows them where a ladder is located or where the mission goal (escape zone/hostages/bomb spot) is. This is done by means of waypoints. You can imagine waypoints a bit like those flags on a ski run. Each waypoint marks a point where bots can go. If two of them are connected with each other, a bot can go from one point to the other and back. So what you do when you waypoint a map is basically place a whole net of points in the map and connect them in a way that bots can proceed from one point to the other. All points must be placed in areas that are accessible for players, and if you want your bots to navigate smoothly and safely, you must also keep an eye on the connections. If connections go through walls or over a deep ravine, your bots will bump into walls or fall to their death. There are several waypoint types that can be used to indicate map goals, rescue zones, nice camp spots, ladders etc. There are different types of connections too, one-way or two-way connections and jump connections that will make a bot jump from point A to B instead of walking or running there. We will come back to this later.

Besides, you don't have to worry about every little detail. The editor that comes with this bot version will do lots of the work for you, and besides, it's graphical and easy to use (no programming/coding skills or anything required). You may very well discover that making waypoints can be fun, especially when you see bots roam through the entire map without problems - and you made it possible!.

Note: Since YaPB 2.10 version, a new waypoint format named **Graph** has been added, which raised the limit to 2048 nodes (in later builds, when graph autoanalysis feature was introduced, the node limit has been raised again), allows you to set camp directions not only horizontally but also vertically, and also reduced the size of waypoint files. YaPB also continues to support the old **PWF** format. You can save waypoints to pwf format, but it will automatically converted to graph format when loaded. Waypoints (multicolored stripes) now named **Nodes** since YaPB 4.x version. Waypoint editor also renamed to Graph editor.

6.1.2 What do waypoints look like in the game?

When you are playing a normal game on a waypointed map, the waypoints will of course be invisible so that they don't distract or annoy you in any way. When waypointing is activated (see: How can I access the waypoint editor?, Below for instructions on how to do that), you will see waypoints as vertical bars about as high as a standing player. The colour of normal waypoints is green, but you may also see waypoints in white, purple, red, blue and cyan. These colours indicate special waypoints, some of which have already been mentioned in the last paragraph. If you see waypoints that are much smaller than the other ones, they are crouch waypoints. They will force bots to crouch when approaching them. Such waypoints are needed to lead bots through vents or any other low and narrow passages. Connections between waypoints are marked as horizontal lines leading from the centre of one waypoint to the other. They, too, exist in different colours. You may see yellow, white and red lines. Don't worry if all these different colours sound confusing right now - it's actually very easy, but of course it helps a lot if you see some waypoints on the screen. Now you know what waypoints and connections are, what they are for and what they look like, you may want to enter the editing mode and see for yourself.

6.1.3 How can I access the waypoint editor?

The waypoint editor is not a separate program, it is included in the bot dll (or .so, if you are using linux). To open it, create a LAN/Listen Server game, select the map you want to waypoint and start the game as usual. As soon as you are in the map, you can activate the editing mode from the console by typing `yb graphmenu` or if you have bound a key for it, simply by pressing that key.

6.1.4 Graph console commands Summary

The following Graph commands are available:

`yb g on`	Turns on displaying of nodes.
`yb g off`	Turns off displaying of nodes.
`yb g on auto`	Turns on auto nodes placement setting (see below).
`yb g off auto`	Turns off auto nodes placement setting (see below).
`yb g on models`	Turns on the player models rendering on spawn points.
`yb g off models`	Turns off the player models rendering on spawn points.
`yb g on noclip`	Turns on nodes editing with noclip cheat. This allows you to fly and you don't collide with walls
`yb g off noclip`	Turns off nodes editing with noclip cheat.
`yb g add`	Adds a node at the current player location. A Menu will pop up where you have to select the diff
`yb g addbasic`	Adds basic nodes on map, like spawn points, goals and ladders.
`yb g cache`	Remember the nearest to player node.
`yb g clean`	Cleans useless path connections from all or single node.
`yb g delete`	Deletes the node nearest to the player (see below).
`yb g erase`	Removes the graph and bot experience files from hard drive.
`yb g flags`	Allows you to manually add/remove Flags to a node.
`yb g setradius x`	Manually sets the Wayzone Radius for this node to value x.
`yb g teleport x`	Teleports player to node index specified in value x.
`yb g stats`	Shows the number of different nodes you did already set.
`yb g fileinfo`	Shows basic information about graph file.
`yb g adjust_height`	Modifies all the graph nodes height (z-component) with specified offset.
`yb g check`	Checks if all node connections are valid.
`yb g load`	Loads the nodes from a graph file (see below).
`yb g save`	Saves the current nodes to a file (see below).
`yb g save nocheck`	Saves the current nodes to a file without validating.
`yb g upload`	Uploads created graph file to graph database.

cont

Table 1 – continued from previous page

`yb g menu`	Show the graph editor menu. Also available via alias `yb graphmenu`
`yb g path_set_autopath`	Opens menu for setting autopath maximum distance.
`yb g path_create`	Opens menu for path creation.
`yb g path_delete`	Delete path from cached (or faced) to nearest node.
`yb g path_create_in`	Creating incoming path connection from faced (or cached) to nearest node.
`yb g path_create_out`	Creating outgoing path connection from nearest to faced (or cached) node.
`yb g path_create_both`	Creating both-ways path connection between faced (or cached) and nearest node.
`yb g path_create_jump`	Creating outgoing jumping path connection from nearest to faced (or cached) node.
`yb g path_clean`	Clears connections of all types from the node.
`yb g iterate_camp`	Allows to go through all camp points on map.
`yb g acquire_editor`	Acquires rights to edit graph on dedicated server. (see below)
`yb g release_editor`	Releases graph editing rights.

To use the graph commands, you will have to use the console. Use the ~ key to bring down the console. Enter the console commands that you wish, then use the ~ key again to return to the game.

6.2 Using graph commands

Using `yb g delete` will remove the waypoint closest to the player. The waypoint MUST be within 50 units from the player (about 1/2 the player height) in order to be removed. You will need to stand fairly close to the waypoint to be able to remove it. This prevents you from accidentally removing a waypoint on the other side of the room. When removing a waypoint you will hear a sound indicating that the waypoint was removed (the same sound the tripmine makes when placed on a wall).

Using `yb g save` will save the waypoint data to the graph file. The graph file will have the same name as the current map with an extension of `.graph`. The file will be saved into the `cstrike/addons/yapb/data/graph` Folder. Your current player name will be saved as the waypoint file author.

You can also save the waypoint in `.pwf` format for older versions of YaPB or PODBot by typing in the console `yb g save old`. Please note that you can save a waypoint in this format only when the number of nodes does not exceed 1024.

Using `yb g load` will clear out all waypoints in the current map and load them from the graph file in the graph folder. This is a good way to "undo" a bunch of waypoints that you have created but do not wish to save. There is no way to `undo` a single waypoint. You will have to use the `yb g delete` command to remove waypoints one-by-one.

The `yb g on auto` command allows you to automatically drop waypoints as you run around in a map. As you run around the level waypoints will be dropped every 200 units automatically. No waypoint will be dropped if another waypoint is already within 200 units of your current position. So if you want to place lots of waypoints fairly close together you may have to manually place some of the waypoints using the `yb g add` command. Autowaypointing keeps track of where the last waypoint was dropped (either manually or from autowaypointing) and will place another waypoint when you are 200 units from the last waypoint. If you don't like where autowaypointing placed a waypoint and want to move it a little bit, you can delete the waypoint using `yb g delete` (but turn off autowaypointing before, since it will place a new waypoint otherwise).

When using autowaypointing, try to stay in the center of narrow hallways and always place a waypoint on BOTH sides of a door. You may have to place some of these waypoints manually using `yb g add` since places like intersections of hallways and doorway entrances and exits don't usually fall exactly at the location where autowaypoint would want to place a waypoint.

Whenever you get close to a waypoint, yellow, white or green lines will be drawn to all of the other waypoints that the bot would consider to be "reachable". If the connection is a two-way connection the line is yellow, one-way connections appear white. These "reachable" waypoints would be waypoints that are clearly visible from the current location. Certain waypoints will be disallowed as reachable for one reason or another. For example, waypoints suspended in

mid-air above the bot would not be considered reachable since the bot couldn't jump high enough to get to them. Also waypoints that are too far away from the current location would not be considered reachable. You may have waypoints that are close enough to each other, but across a wide gap that would be too wide to jump. If the far waypoint is close enough and clearly visible, it would still show as "reachable" since currently we have no method to determine if the bot can get to that waypoint or not.

The bots will ONLY go from one waypoint to another if there is a path between them. Get in the habit of checking that paths exist BOTH WAYS between waypoints. Just because a path is drawn from point A to point B, doesn't mean that a path exists from point B to point A.

The `yb g path_create` command allows you to manually assign a path between 2 waypoints. This is needed in some cases where the waypoints are blocked (by doorways or other objects) and you wish to create a path between these waypoints. Move close to the waypoint you wish the path to start from and use the menu to add path.

The actual Waypoint Number you're standing on will be shown in the upper corner of your HUD. For example to manually assign a path between Waypoint #250 and 251, you first should stand in the near of #250, then use `yb g cache` to cache waypoint #250, then go to the waypoint #251 and type `yb g path_create` to show menu and create needed path connection (one-way or two-way). You can also do this by looking at needed waypoint instead of caching it.

The `yb g path_delete` command is just like the "create" command except that it removes a path (connection) from the starting point to the ending point. This is necessary in some cases where you may have a door that opens from one side and allows you to go through but once the door closes you can't go back through the other way.

Using the `yb g path_clean` command will remove all connections of any type from the nearest node, unless a node index number is specified as an argument.

The `yb g acquire_editor` command allows you to edit graph on dedicated server. Before you can use this, the `yb_password` and `yb_password_key` cvars must be configured on the server.

6.3 Installing waypoints

By default, if YaPB finds a graph (waypoint) in the official database for your map, it will automatically download it to the `addons/yapb/data/graph` folder.

If you want to install graph manually, put it in the `*gamedir*/addons/yapb/data/graph` folder.

Or if you want to install the old format (pwf) waypoint manually, put it in the `*gamedir*/addons/yapb/data/pwf` folder.

Also you can install the E-BOT (ewp) waypoints in the `*gamedir*/addons/yapb/data/ewp` folder.

Where `*gamedir*` is the path to the game directory, for example:

- `D:\Steam\steamapps\common\Half-Life\cstrike` is the Counter-Strike 1.6 folder.
- `D:\Steam\steamapps\common\Half-Life\czero` is the Counter-Strike Condition Zero folder.

6.4 Graph Editor overview

The Graph Editor shows some useful information about Graph and Practice data, such as properties of current/faced/cached node, node practice data, map name and your current time.

Graph data are stored in `.graph` file at `addons/yapb/data/graph` or `.pwf` file at `addons/yapb/data/pwf` if you using/saving it to old PODBot waypoint format. Practice data are stored in `.prc` file at `addons/yapb/data/train` folder.

Current/Faced/Cached node information shows index number of this node, total amount of nodes, radius, light value, flags and origin. If you didn't cache any node or you are not currently facing any node at all, there will be only the data displayed of nearest node.

Node practice data shows index number of node and the damage value taken from it for both T (Terrorists) and CT (Counter-Terrorists). You can also see arrows pointing to these nodes (red for Terrorists, blue for Counter-Terrorists).

White arrows are pointing to your faced node (on which you point your crosshair). Yellow arrows are pointing to your cached node.

Fig. 1: Graph Editor viewport.

6.5 Adding waypoints

Adding a waypoint is really easy. Just walk to the position where you want a waypoint to be inserted, bring up your waypoint menu:

To add a waypoint, simply select `5. Add waypoint`. A new menu will appear, the "Waypoint Type" menu. All waypoint types described below can be added by using this menu.

Once you have selected a waypoint type from the "Waypoint Type" menu, you will hear a sound, and the selected waypoint will appear in the map, at the exact position where you stood when you pressed the key.

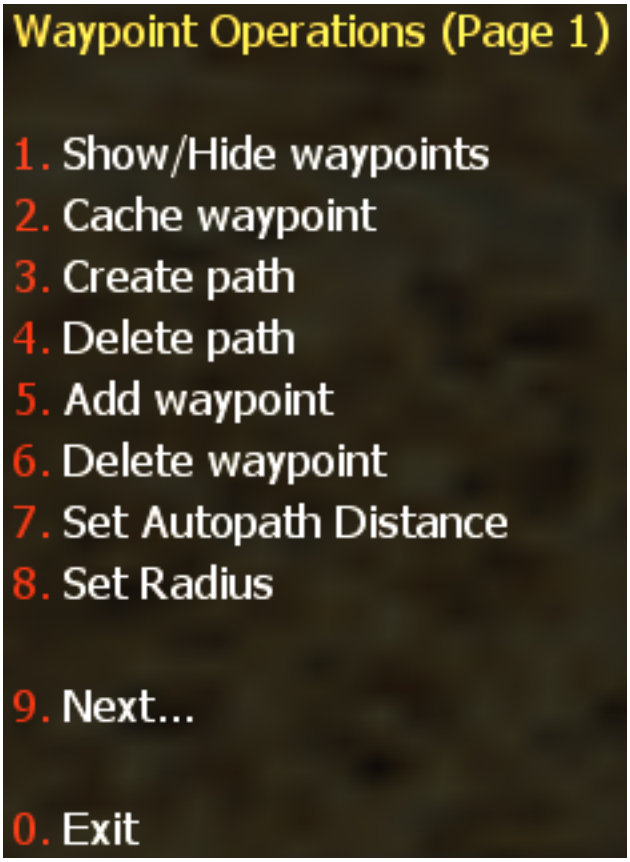

Fig. 2: Waypoint Menu (Page 1).

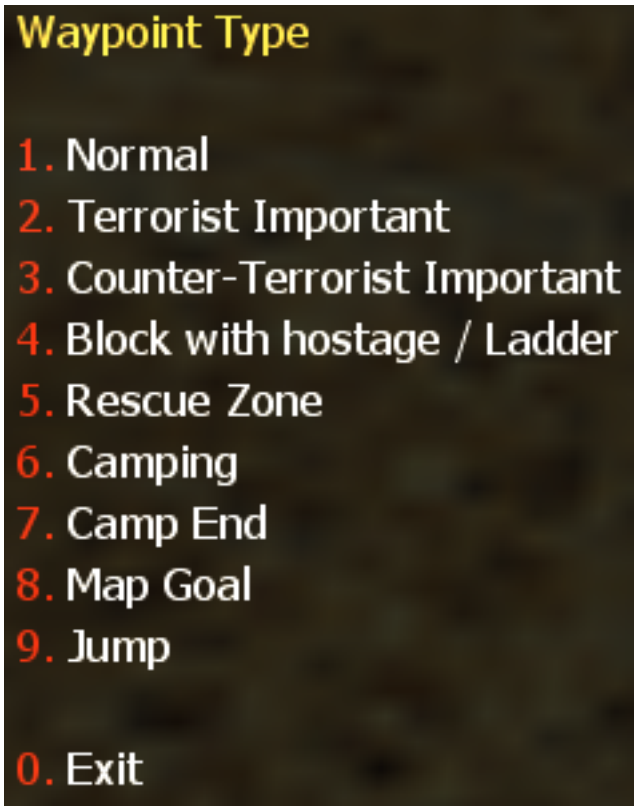

Fig. 3: Waypoint Type Menu.

Note: If you are standing while selecting a waypoint, a standing waypoint will be inserted. All bots will run or walk towards this waypoint normally. If you want bots to crouch when approaching a particular position, crouch down when inserting the waypoint. You will notice that the waypoint you just added is only about half as high as normally. As you inserted it when you were crouched down, it automatically carries a "Crouch" flag (see: Types Of Waypoints). Bots will now crouch automatically when trying to reach this waypoint.

Now that you know the basic method used to add a waypoint, let's have a closer look at the waypoint types that exist.

6.6 Types of Waypoints

6.6.1 Normal waypoints

Normal waypoints are the points you need in order to make bots walk through the map. They are used for navigation only and will not trigger any particular behaviour. You can add a Normal waypoint by selecting `1. Normal` from the "Waypoint Type" menu. The colour of Normal waypoints is green, as you can see in the picture below.

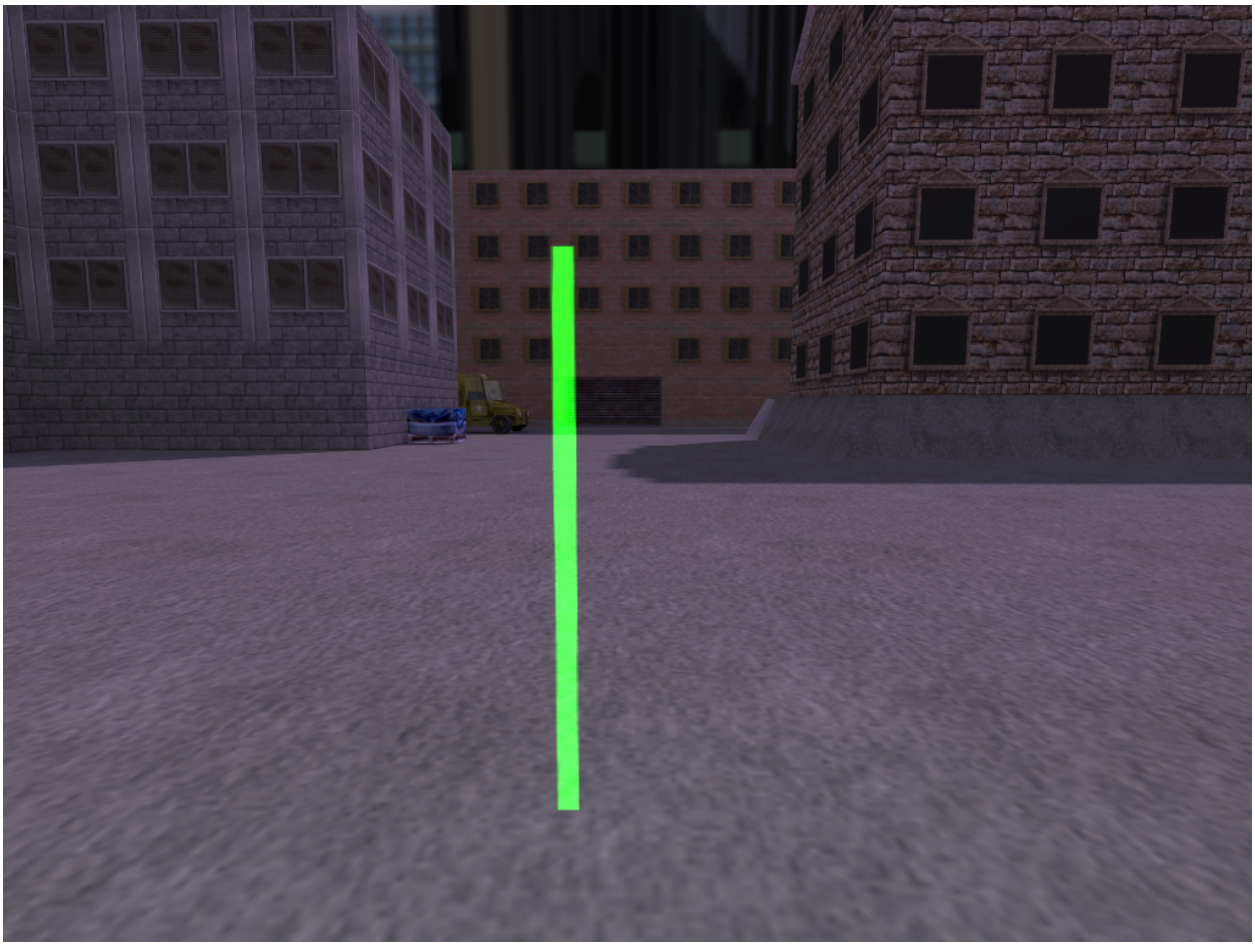

Fig. 4: Normal Waypoint.

6.6.2 Terrorist Important waypoints

This type of waypoints can be navigated just as a Normal waypoint by all bots, but it has one additional function. It marks strategically important points for a Terrorist team. Adding a Terrorist Important point in a room will tell Terrorist bots to go to the room and check it frequently. You can add this type of waypoint by selecting `2. Terrorist Important` from the "Waypoint Type" menu. The colour of Terrorist Important waypoints is green with red head, as you can see in the picture below.

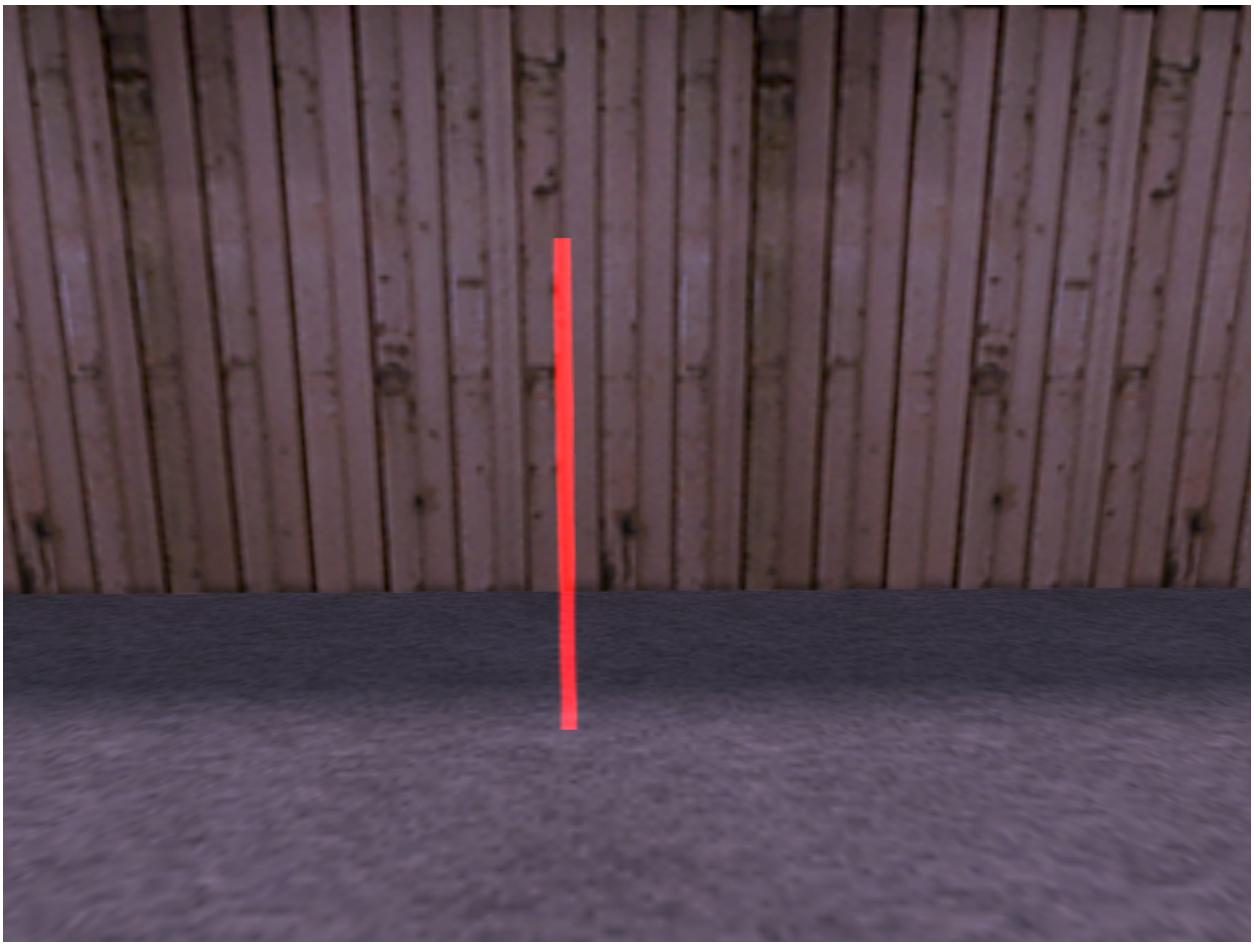

Fig. 5: Terrorist Important Waypoint.

Important: The use of Terrorist Important points depends on the map type! Wherever the Terrorist team is the "defending" team (i.e. on **As_** **Cs_** type maps), Terrorist Important points should be placed at key positions around the hostage area or VIP escape zone. For example, if the hostages are inside a building, Terrorist Important points should be added behind each entrance to the building. Doing so will make the Terrorists check all entrances frequently and guard them. Do not place Terrorist Important points far away on the other side of the map. After all, you don't want the Terrorists to abandon the hostages and rush planlessly through the map, now, do you? With the VIP escape zone, the same strategy applies, Make Terrorists guard the key routes to the escape zone by using Terrorist Important waypoints. You DON'T need to place Terrorist Important points directly at the hostages. Terrorists will check on hostages anyway. On maps where the Terrorist team is "offensive" (i.e. **De_** and **Es_** type maps), Terrorist Important waypoints should not be overused. The "offensive" team will try to reach the map goal waypoint anyway. The only useful function you can use important waypoints for is to make particular routes more attractive for the bots. For example, if there is a longer and more complicated, but safer and more surprising route to the map goal, bots may tend to underuse it a little.

In such cases, placing one or two Terrorist Important waypoints along this route can help.

6.6.3 Counter-Terrorist Important waypoints

The function of this waypoint type is exactly the same as the Terrorist Important waypoint described above. The only difference is that a Counter-Terrorists Important waypoint obviously marks strategically important places for the Counter-Terrorist (CT) team. You can add this type of waypoint by selecting `3. Counter-Terrorist Important` from the "Waypoint Type" menu. The colour of Counter-Terrorist Important waypoints is green with blue head, as you can see in the picture below.

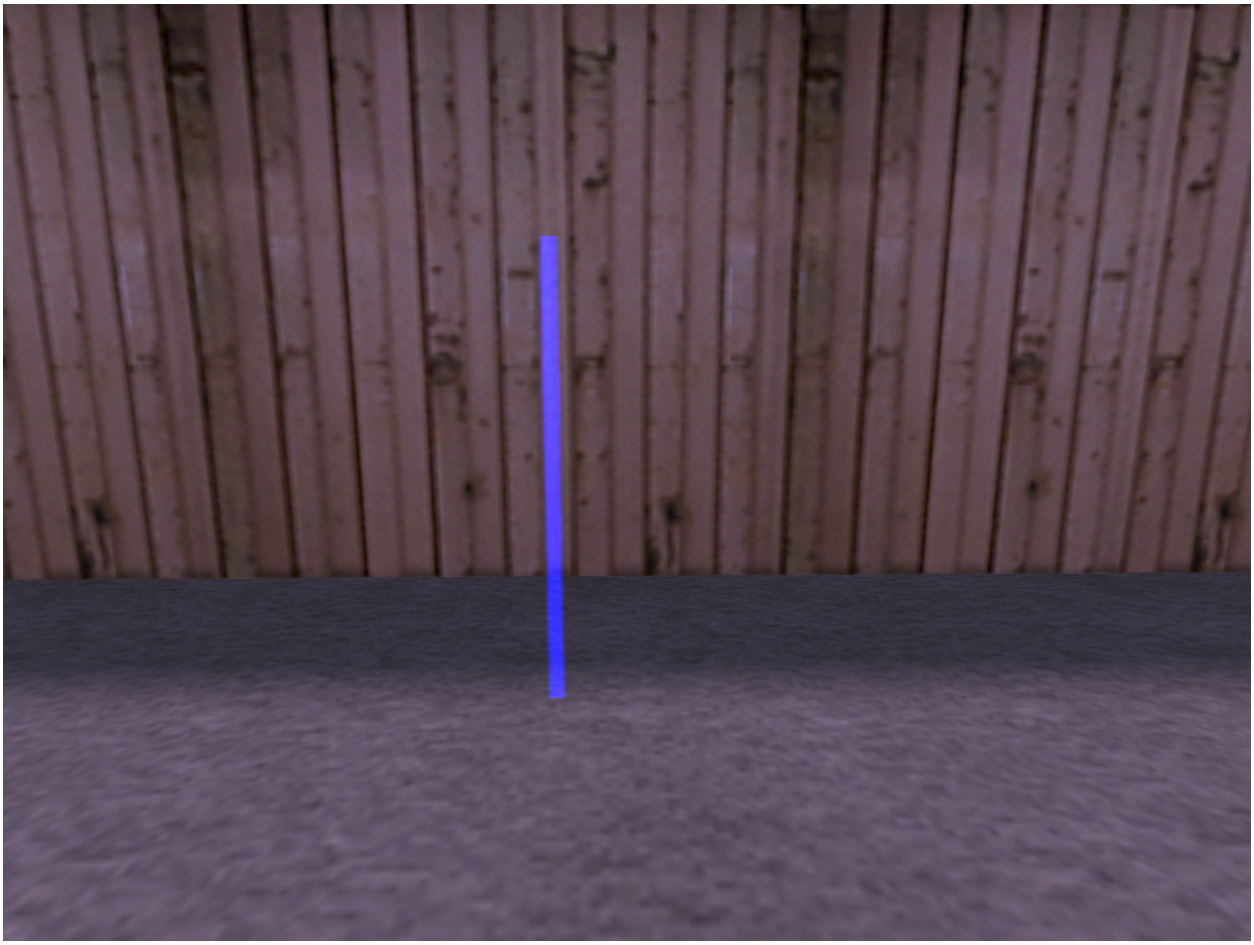

Fig. 6: Counter-Terrorist Important Waypoint.

Important: As with the other team specific waypoints, Counter-Terrorist Important waypoints should also be placed according to the map type. On maps where the Counter-Terrorist team is forced to move out and reach a certain goal - either hostages to rescue or a VIP escape zone to reach safely. Counter-Terrorist Important points can be useful to make a particular route more attractive. You DON'T need to place Counter-Terrorist Important points near a map goal (hostages on **CS_** maps, VIP escape zone(s) on **As_** maps), Counter-Terrorist bots will go there anyway. It's the most important point for them, and adding several other important waypoints right next to it doesn't yield any benefit. On maps where the Counter-Terrosist team is in a defensive role (i.e. on **De_** maps and **Es_** maps), place Counter-Terrorist Important points at key positions around the bomb/escape zone(s) in order to make Counter-Terrorist bots defend all

possible routes to the Terrorists map goal.

6.6.4 Ladder waypoints

Ladder waypoints are only used for waypointing ladders, as you possibly guessed. To enable your bots to use a ladder, simply walk up to the ladder until you get "stuck" on it (you will see your crosshair grow wider once you are on the ladder). Now place one Ladder waypoint at the bottom of the ladder. Then climb up the ladder until you are almost completely over the edge. Place a second waypoint here and make sure that the two ladder waypoints are connected, (This should have happened automatically if the Ladder waypoints aren't too far away from each other, if not you can create a connection manually), That's all! You can add this type of waypoint by selecting `1. Normal` from the "Waypoint Type" menu, it will automatically turn into a ladder waypoint if you are standing on ladder. Or choose `4. Block with hostage / Ladder` from the "Waypoint Type" menu if it's a Hostage Rescue (CS_) scenario map so that bots don't miss the hostages when going up on ladders. The colour of Ladder waypoints is brown, as you can see in the picture below.

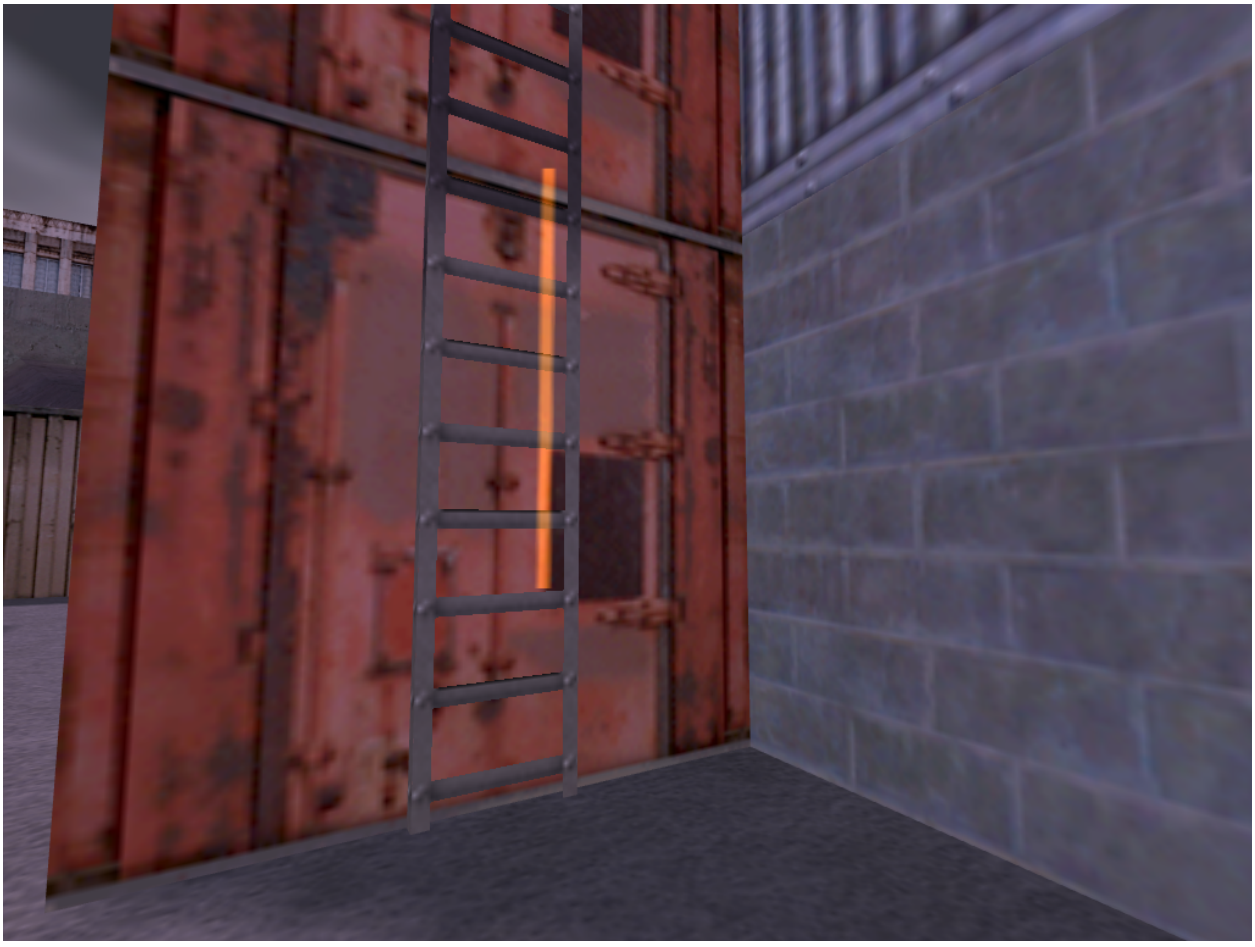

Fig. 7: Ladder waypoint

Some general hints and notes concerning ladder waypoints:

1. Waypoint ladders AFTER you waypointed the areas above and below them! If you waypoint ladders first, all waypoints in reach of a ladder waypoint will be connected with it and have their radius reduced to zero automatically! It doesn't matter whether you place the top or the bottom Ladder waypoint first.
2. If the ladder is very long, you can place additional Ladder waypoints between the bottom and the top end.
3. The bottom waypoint will automatically get connected with the nearest waypoint, independent of current AutoPath Max Distance settings.
4. The top waypoint will usually get a connection towards it automatically, but you will have to add a connection leading away from it manually.
5. Ladder waypoints will always have a radius of zero, and this shouldn't be changed!

6.6.5 Rescue waypoints

Rescue waypoints are only needed on Cs_ type maps (hostage rescue scenarios). They mark the zone where the Counter-Terrosist team must bring the hostages, the rescue zone. Place one of these waypoint inside each rescue zone there is. If there is only one, you only need one **Rescue** waypoint. Placing more points in one rescue zone is unnecessary bulk and will rather cause problems than improve anything. A Counter-Terrorist bot that has succeeded in "activating" the hostages will determine the position of the nearest rescue point and lead the hostages there. When the bot has reached the rescue point, it will check if the hostages are really rescued and after max time about 5 seconds turn back to return to combat. Badly placed rescue points may lead to bots turning around before the hostages have really reached the rescue zone. As a consequence, the hostages will be left standing a few inches away from the rescue zone while the bot considers its mission as completed and turns back to fight, ignoring the deserted hostages. That's why you are advised to place a rescue waypoint well inside a rescue zone, not at its edges! In the editor, rescue points will be displayed in bright white (see below). Their radius is set to zero by default and shouldn't be changed. All bots can use this waypoint type for Normal navigation as well. You can add this type of waypoint by selecting `5. Rescue Zone` from the "Waypoint Type" menu. **The colour of Rescue waypoints is white**, as you can see in the picture below.

6.6.6 Camp waypoints

As the name suggests, Camp waypoints are used to mark good sniper spots. They can be navigated by all bots. However, whether a bot may camp there or not is determined by the flag you can add to the camp waypoint. You can make Camp waypoints team specific or leave them "open" to any team. The colour of Normal Camp waypoints is cyan. Terrorist specific camp waypoints have coral color, Counter-Terrorist specific is cornflower blue color, as you can see in the picture below.

Although there are two entries in the "Waypoint Type" menu ("Camping" and "Camp end"), the Camp waypoint is in fact only one point. However, it carries two "markers" that tell a camping bot where to look while camping. When you are camping yourself, you will monitor a certain area. If you wanted to define this area, you could describe it as an angle. This angle would be specified by two lines going out from your position: One that marks the left edge and another one for the right edge. The monitored area would be between these two lines. The mentioned "markers" fulfill exactly this function. They are displayed as more or less horizontal beams going out from the top of a Camp waypoint. The colour of Camp markers is red, as you can see in the picture below.

When a bot approaches the depicted Camp waypoint, it will turn to face the direction of the Camp start marker first. Then it will scan the area between this marker and the Camp end marker by changing every few seconds the direction it is facing from one to the other. An enemy moving outside the two markers may escape the bot's attention, unless it hears the enemy coming. In the piture above, both markers are pointing to the same height. However, you can also specify different heights for each marker. This is very useful for making bots monitor a ramp, a slope, a stairway or other uneven surfaces. So far, so good. But how to set a working camp waypoint? Follow these steps:

Fig. 8: Hostage Rescue waypoint

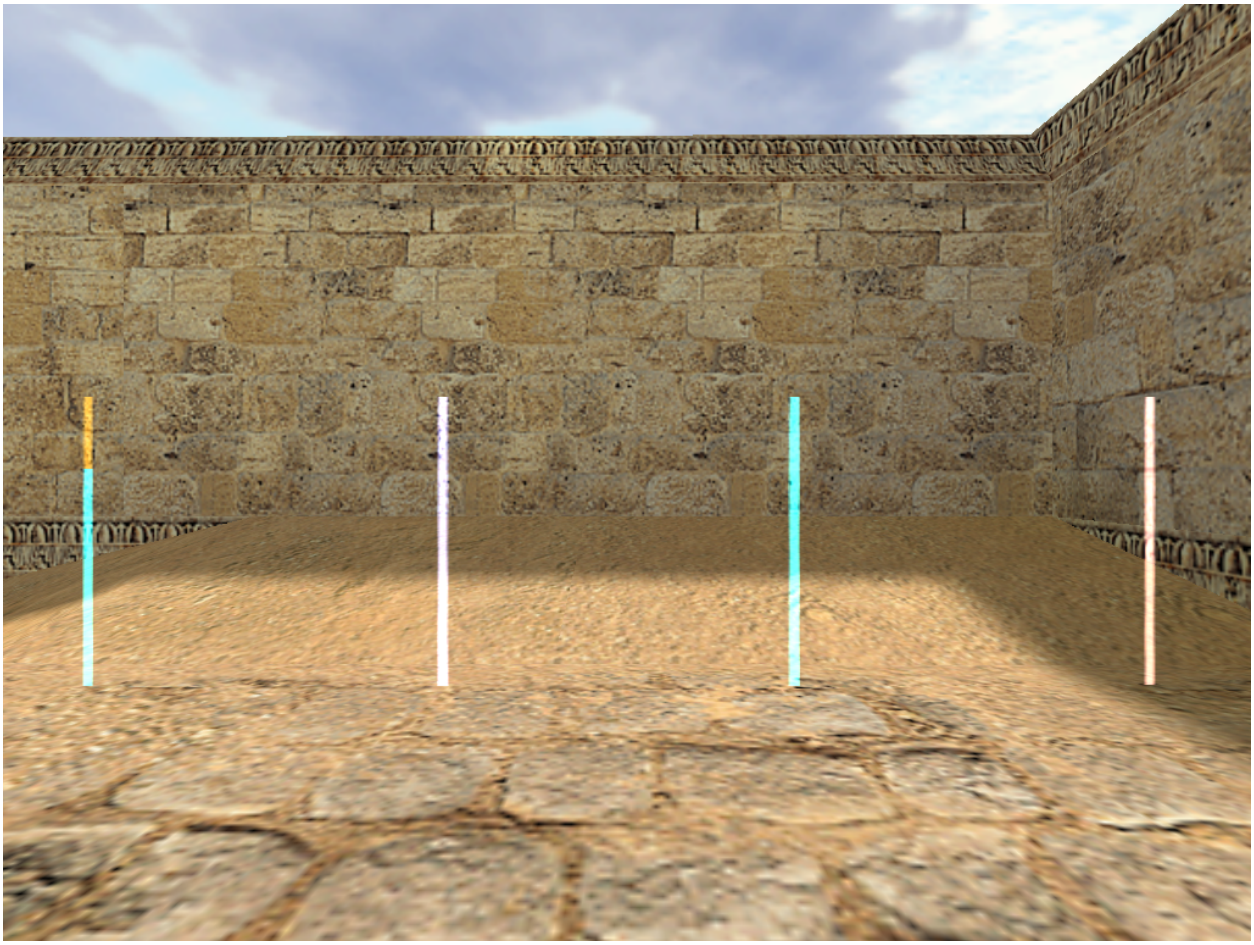

Fig. 9: From left to right. Sniper, Counter-Terrorist Specific, Normal and Terrorist Specific camp waypoints.

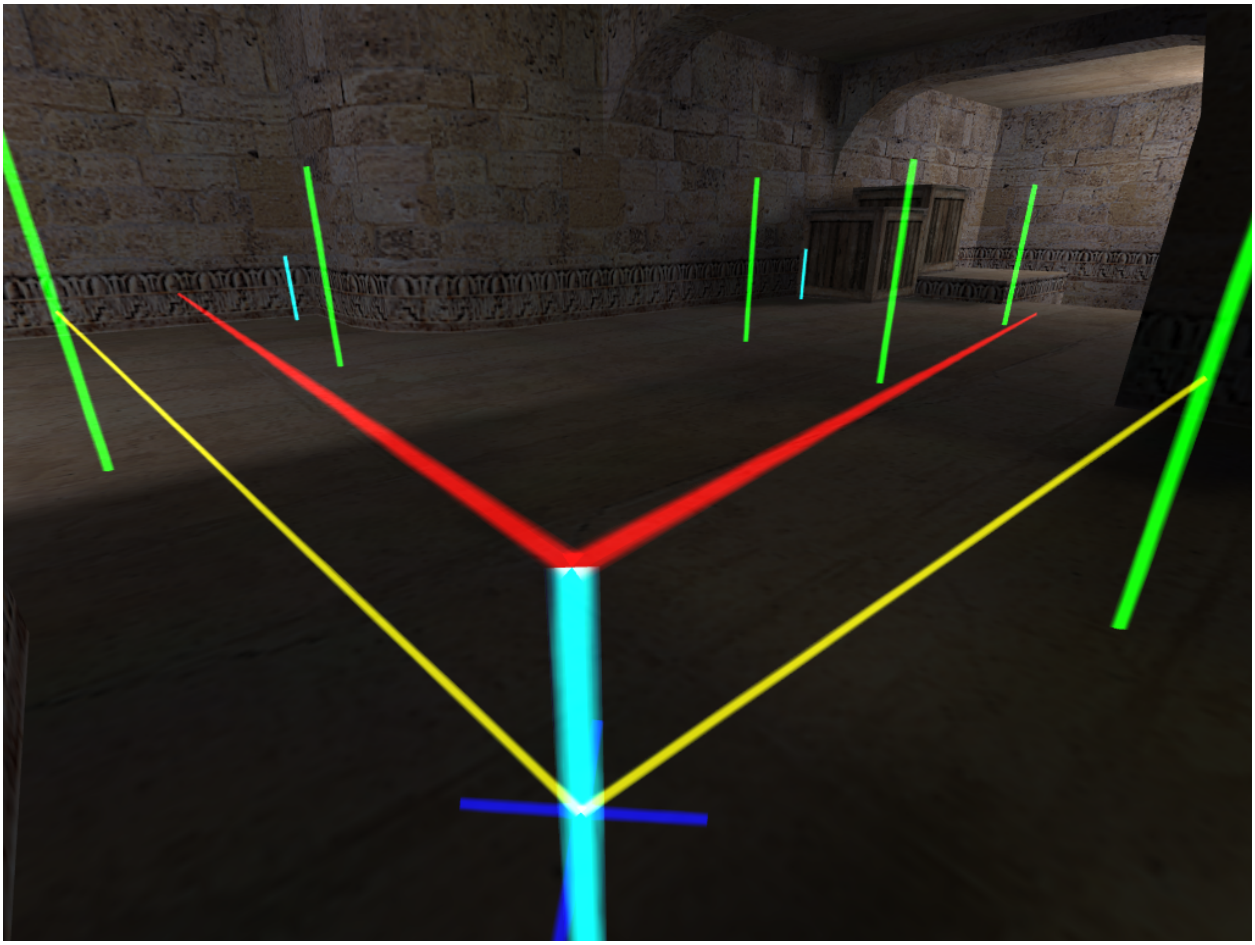

Fig. 10: A crouched normal Camp waypoint with Camp start and Camp end markers (directions)

1. Go to the exact position where you want bots to camp (of course, a dark corner or similar locations are best suited for camping, but whom do I tell this?)
2. If you want bots to stand while they are camping, remain standing upright. If you want them to crouch while camping (more precise aiming!), crouch yourself while adding the point
3. Point your crosshair at the exact direction and height where you want your bots to start looking.
4. Bring up the "Waypoint Type" menu and select `6. Camping`. The Camp waypoint itself will now be placed at your current position, and you will see the two marker beams going out from it. The Camp start marker will already be pointed at the direction you specified, the Camp end marker will still need some adjustment.
5. Now point your crosshair at the exact direction and height where you want your bots to end their monitoring.
6. Once again, open the "Waypoint Type" menu, but now select "7. Camp end". You will see that the Camp end marker will now be pointed at the direction you specified.

That's it! Unless you want to make your Camp waypoint team specific or add another flag (see: Waypoint Flags section), you are done! In fact, it sounds much more complicated than it actually is.

Some quick notes and hints about Camp waypoints:

1. You can alter Camp start and Camp end markers as often as you want. As soon as you are near an existing Camp waypoint (i.e. as soon as its waypoint stats are shown in the upper left corner of your HUD), Bring up the "Waypoint Type" menu and selecting `6. Camping` or `7. Camp end` will NOT add a new waypoint. Instead, it will readjust the Camp start and/or Camp end marker(s) of the nearby Camp waypoint to the new direction you specified.
2. Thus, if you want to place two Camp waypoints closely together, make sure that the waypoint stats of the first one have disappeared from your HUD before you set the second one. If the stats of the first waypoint are still visible, you will accidentally modify the Camp start and Camp end markers of that waypoint instead of inserting a new point.
3. Don't place Camp waypoints in strategically irrelevant areas, or you will see bots having a situation totally unimportant areas while their team mates are under heavy attack.
4. Provide the "defending" team with some nice sniper spots near the map goal! In general, if you make team-specific Camp waypoints, make more for the defending team than for the attacking team.

6.6.7 Map Goal waypoints

This waypoint type obviously indicates the Map Goal.

On an **As_** map, the Map Goal waypoint tells the bots where the VIP escape zone is. Make sure the escape zone symbol is visible on your HUD when you place a map goal waypoint there. Otherwise the VIP may end up reaching the point and running away again just like you would do with **Rescue** waypoints. On a **Cs_** map, the Map Goal waypoint marks the position of the hostages. It is NOT necessary to place one Map Goal waypoint per hostage. Unless the hostages are standing really far away from each other, one point per hostage group will perfectly do. On a **De_** map, the Map Goal waypoint marks the bomb spots. It must be placed somewhere inside the bomb zone, i.e. the bomb icon must be blinking on your HUD when you place such a waypoint. In contrast to **Cs_** maps, on **De_** maps it makes sense to set various goal waypoints in one bomb zone. This will enable bots to choose from several spots to plant the bomb and make them less predictable.

Note: On **Es_** maps, the Map Goal waypoint marks the escape zone for the Terrorists. You can follow the same rules as for **As_** maps. Now you may wonder how to determine the exact function of a the Map Goal waypoint. Don't worry, this is entirely map specific, you don't have to do anything about it. All bots of both participating teams will automatically know what the the Map Goal is, they only need the point to guide them there. You can add this type of

waypoint by selecting `8. Goal` from the "Waypoint Type" menu. The colour of Goal waypoints is purple, as you can see in the pictures below.

Examples of Goal Waypoints

Fig. 11: Hostage Goal Waypoint (map cs_assault)

Fig. 12: VIP Escape Goal Waypoint (map as_oilrig)

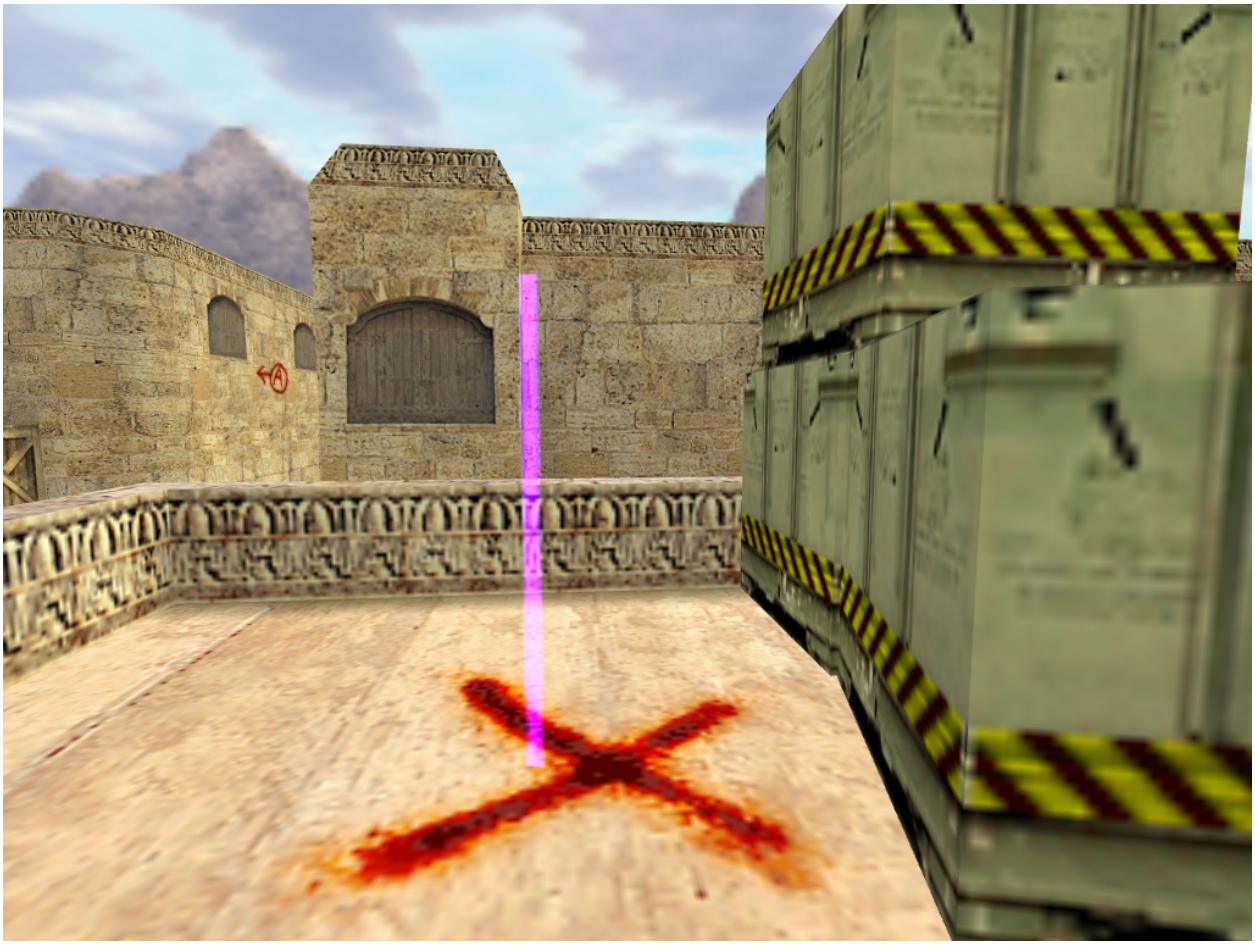

Fig. 13: Bomb place Goal Waypoint (map de_dust2)

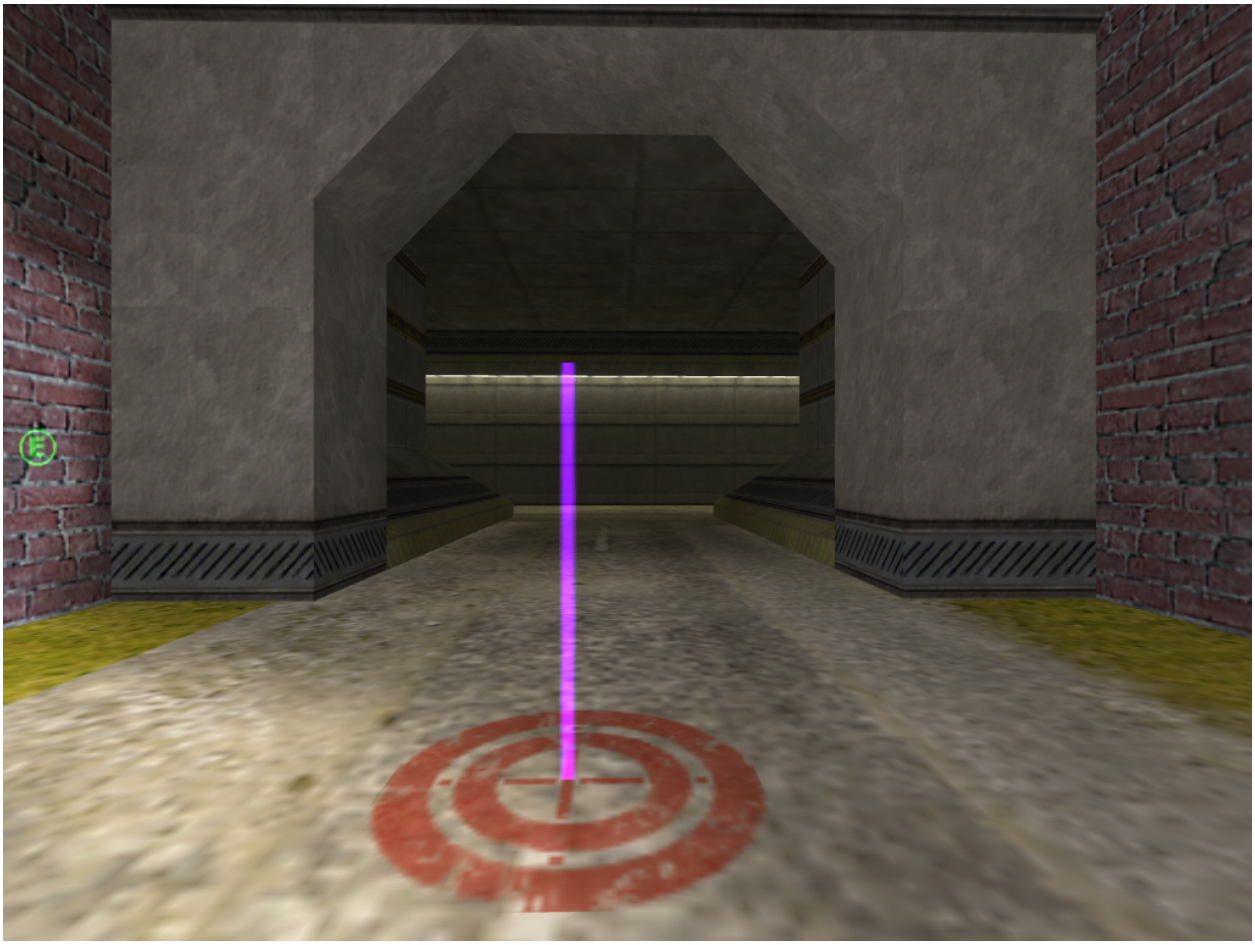

Fig. 14: Terrorists escape Goal Waypoint (map es_trinity)

6.6.8 Jump Connections

Jump is not actually a waypoint but a connection between waypoints. (see: Waypoint Connections)

So now we've seen what type of waypoints the bots use, we can see how we can string these waypoints together to make a giant web to cover the map with.

6.7 The Radius

The Radius of a waypoint is indicated by blue polygon that go out in all directions from the position of a waypoint. The photo below was taken from above, we are looking down onto a waypoint, and you can see the colour of a Radius is Blue, as you can see in the picture below.

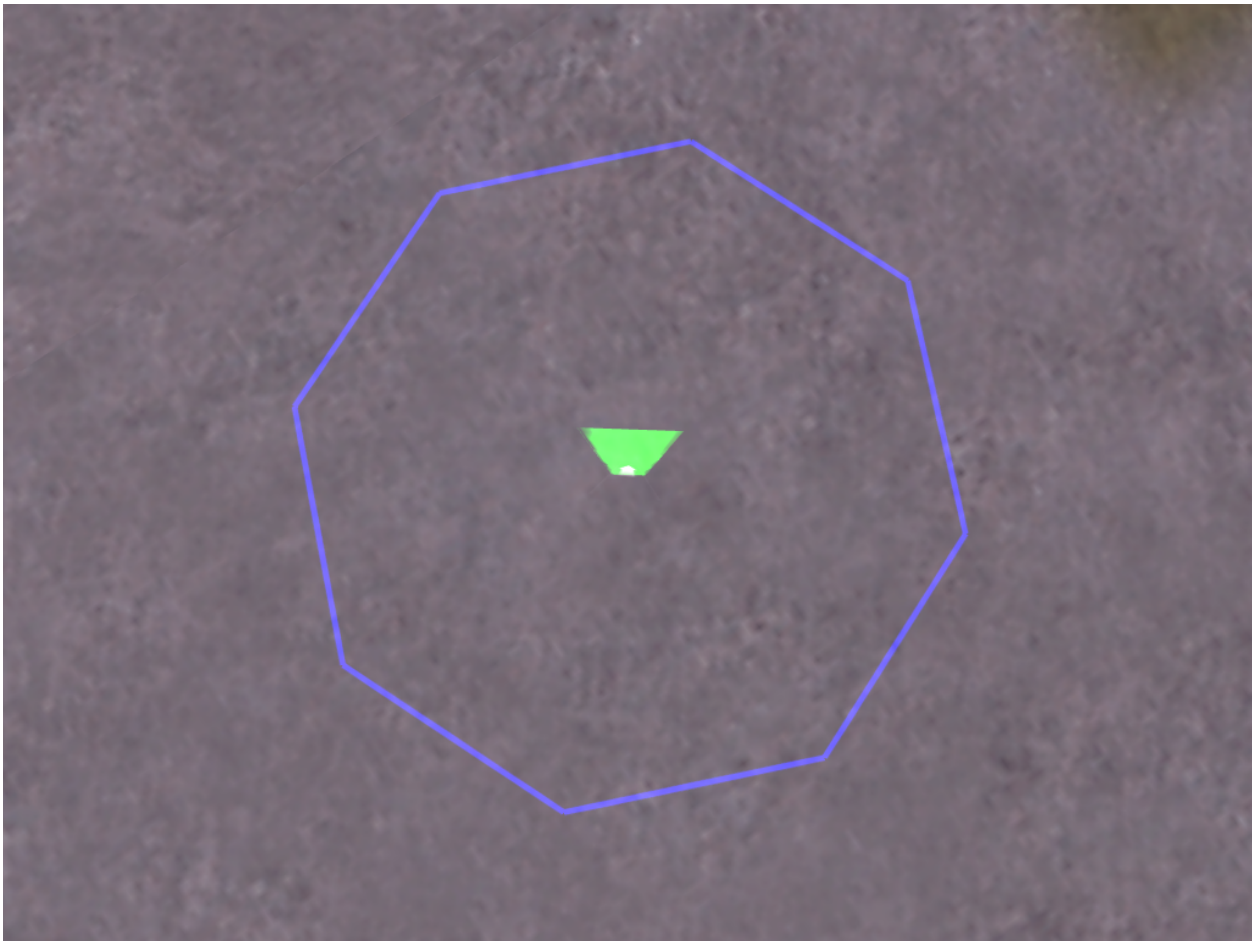

Fig. 15: Waypoint Radius

The Radius indicator are a great mean to see how big the radius of a waypoint actually is, it ends exactly where the corners end. What does the Radius actually do? It tells the bots how exactly they must navigate around the waypoint in question. If a bot walks past a row of waypoints with big radius, it will know that super-exact navigation is not required. If the radius are small, the bot will stick strictly to the waypoints. Thus, in open areas big radius help to make bot navigation natural - you wouldn't want to see a bot run across a wide courtyard as if it was following an incredibly thin, straight line painted onto the ground, now, would you? It looks much more natural if the bot takes advantage of the space he has around him. However, in narrow corridors and doorways or on bridges, the situation is different, too big

radius would make bots too careless, they would bump into walls or even fall off a bridge because they think they can walk anywhere inside that big radius! That's why choosing appropriate waypoint radius is so important. As a general rule, keep the radius big in open areas and make them small in narrow passages. Read the following section to learn how:

6.7.1 Setting or Changing the Radius:

The good news first: You don't have to set every single Radius manually, the editor will do much of the work for you! It will automatically calculate the Radius of a waypoint depending on the area around it. If the editor detects higher (~more than knee-high) obstacles like walls nearby, it will automatically adjust the Radius to reach up to the wall, not further. However, the maximum Radius is limited to 128 units. This means that even on a totally open plain, where the nearest obstacle is hundreds of distance units away, the Radius will not exceed 128.

Now you might wonder "Well, if the editor does all this for me, why would I change radius by hand?" The answer is simple,the editor helps, but it isn't perfect, it cannot detect all kinds of obstacles (I can't go too much into detail here because all this is pretty much map-related). Anyway, you will see places where the radius cuts into an obstacle - it could be a very thin pillar, a fence, or even a solid-looking car that's parked in the streets… as I said, it depends on how the mapper built these elements. Another problem is not with walls, but with holes and cliffs, If there is no high obstacle, the editor will regard an area as clear and set a big radius, it doesn't care if right next to the waypoint there's a yawning abyss where bots will fall to death! So these are areas where you will have to keep a watchful eye on your radius and if necessary, change them by hand. In narrow corridors and especially around narrow doorways, you will see that even a small radius calculated by the editor doesn't make the bots navigate precisely enough. In such places, I strongly recommend lowering the radius to zero. In order to change the radius of a waypoint, bring up the waypoint menu and select "8. Set Radius". The following menu will appear:

Fig. 16: Waypoint Radius Menu

Simply select an option by pressing the corresponding key, and the radius of the currently active waypoint will be changed to the selected value. You will quickly get a feeling for these numbers if you play around with them a bit.

Waypoints with fixed radius

Note: Some waypoint types will always have, and require, a radius of zero. The radius of these waypoint types should NOT be changed! The types with fixed radius are: Ladder, Rescue, Camp (no matter if team specific or not) and Map Goal.

6.8 Connecting Waypoints

Waypoints alone aren't sufficient to make bots move the way you want. They must be connected with other waypoints in order to let bots reach their goal. By default, connections up to a certain distance will be made automatically. You can select the automatic connection distance (AutoPath Max Distance, APMD) by bringing up the waypoint menu and selecting "7. Set Autopath Distance". The following sub-menu will appear:

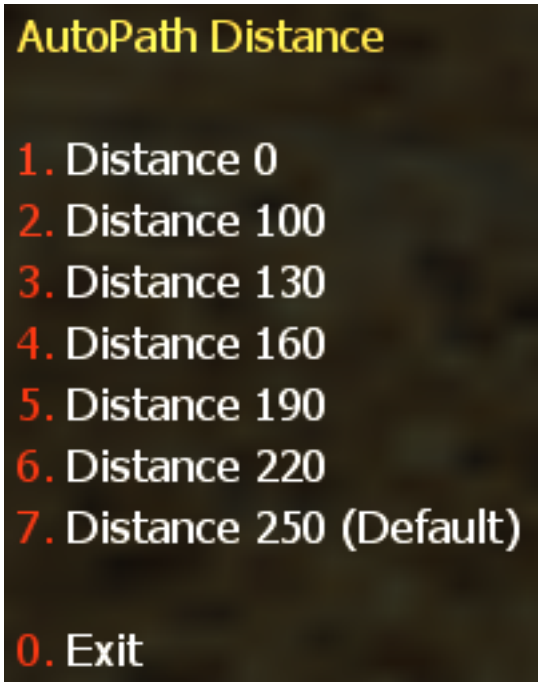

Fig. 17: Autopath Distance Menu

Select the desired distance from this menu. After selecting a distance, connections up to that distance will be drawn automatically, of course, you can also add and remove connections manually. However, this needs a little bit more explanation, and it's better to explain if you know the different connection types. So let's first have a look at these - afterwards you can read how to add and remove such connections by hand.

6.8.1 Two-way (bidirectional) connections

The vast majority of all connections in a waypoint set will be bidirectional. Obviously, these connections enable bots to walk both from point A to point B and back from point B to point A. The colour of bidirectional paths is yellow, as you can see in the picture below.

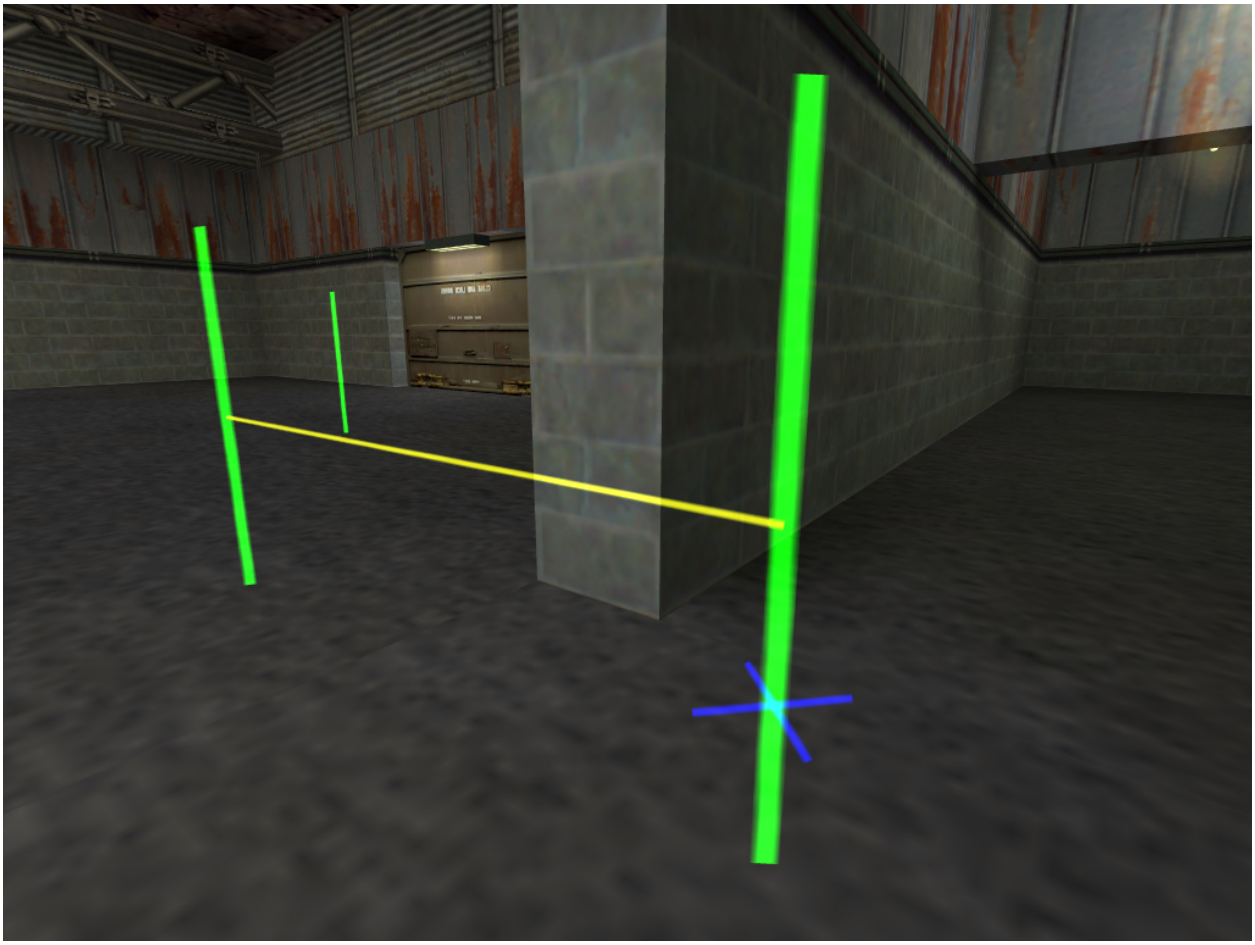

Fig. 18: A 2-Way connection between the two depicted waypoints.

6.8.2 One-way connections

One-way connections enable bots to walk from point A to point B, but not vice versa. They can be useful to make bots drop down a wall or a high crate, but prevent them from attempting to get up. Of course, there may be more places at which a 1-way connection can make sense, but that depends on the map. In game, 1-way connections will be visible from 2 waypoints, their start and their end waypoint. In order to show you the direction of 1-way connection, it will be shown in different colours depending on from which perspective you view it. Let's say you have a 1-way connection from point 1 to point 2. In this case, when you are standing at point 1, you will see the outgoing 1-way connection displayed in white as you can see in the picture below.

The picture below shows the same two waypoints with the incoming connection (from left to right waypoint), The incoming 1-way connection displayed in teal as you can see in the picture below.

Note: If you created an outgoing path connection from waypoint A to waypoint B, it will be displayed in white. And

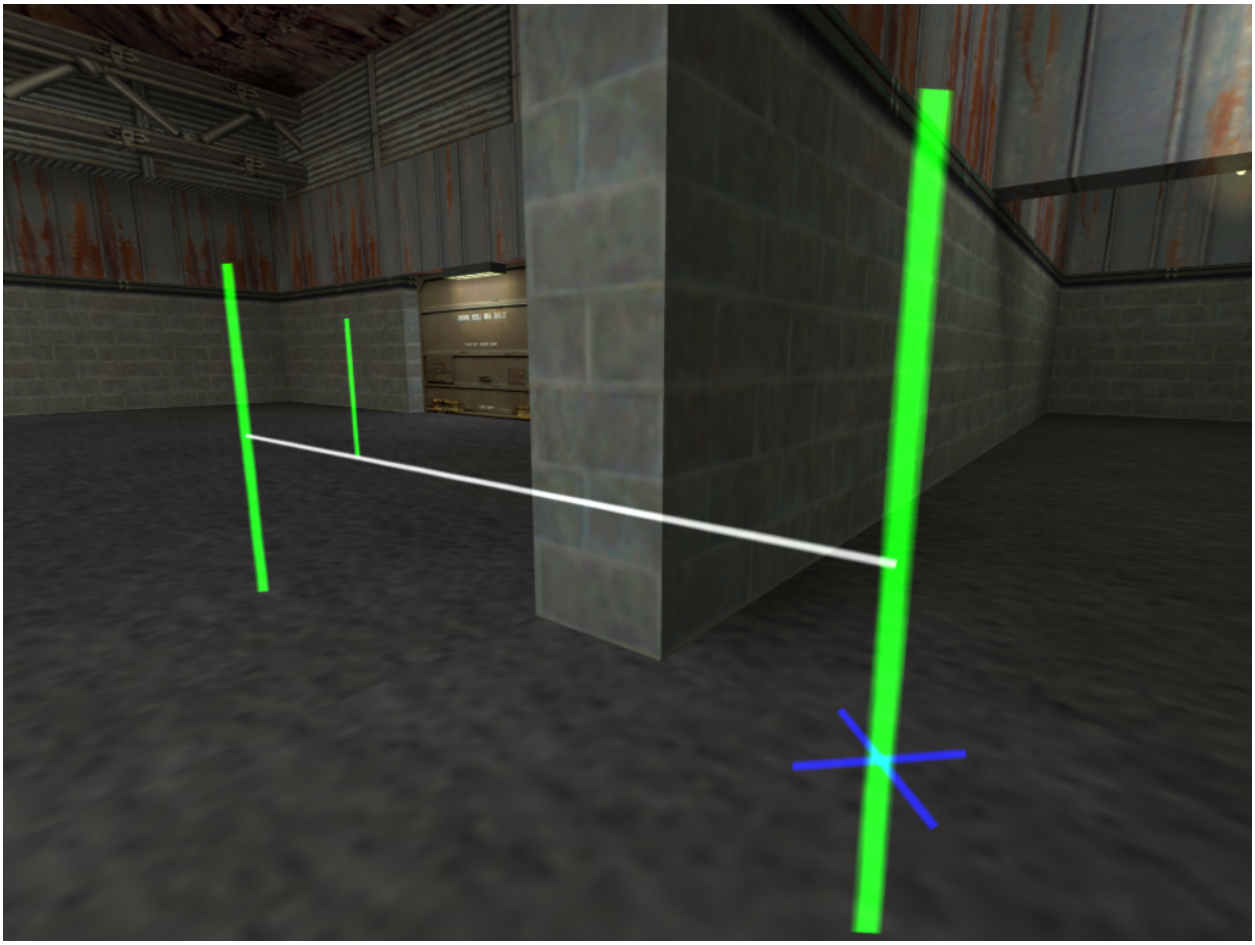

Fig. 19: An Outgoing 1-Way connection from right to left waypoint.

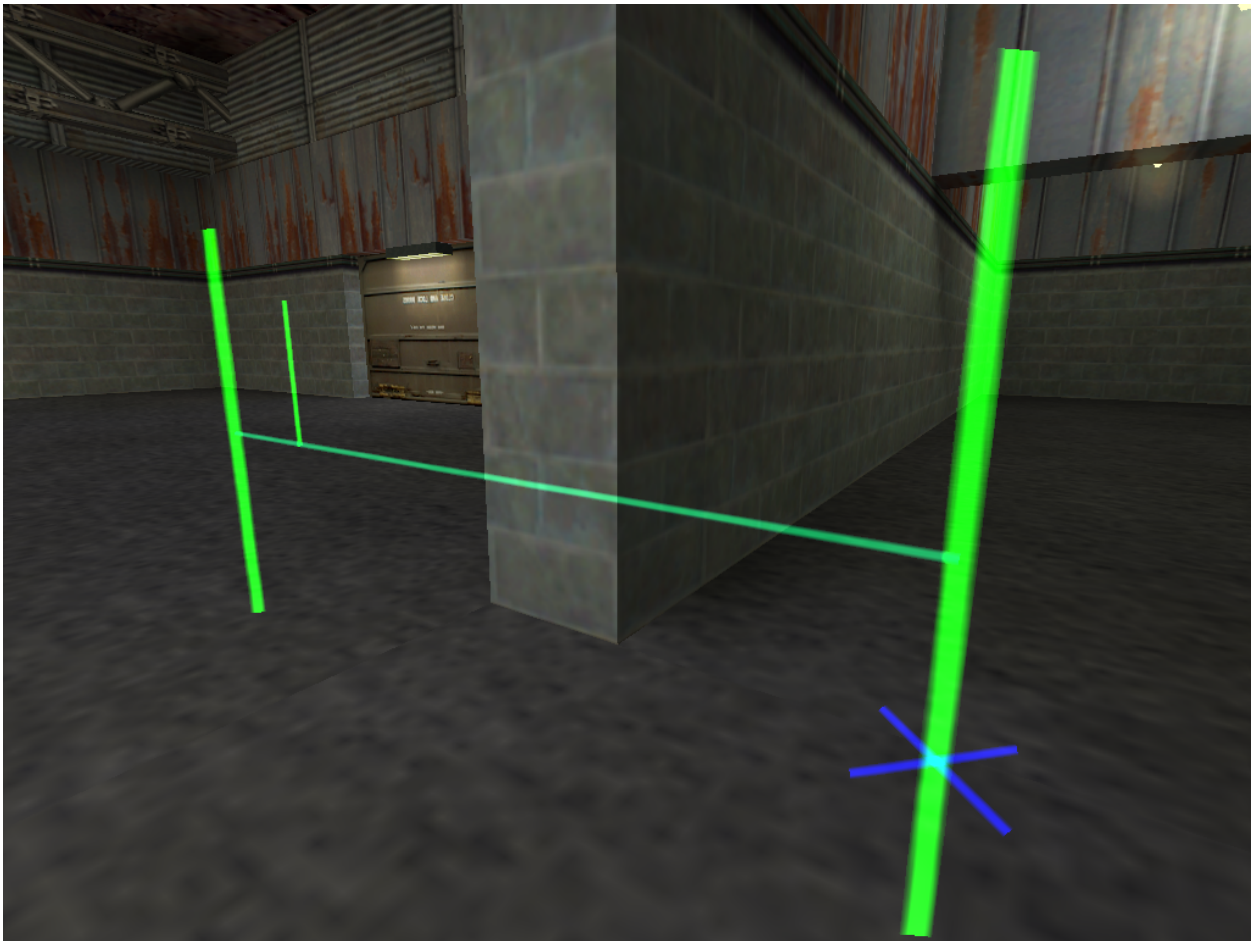

Fig. 20: An Incoming 1-Way connection from left to right waypoint.

when you get to waypoint B, the path connection will turn teal like an incoming path connection.

The fact that 1-way connection are shown from both involved points is a great feature. It makes spotting errors very easy and saves you the trouble of running around to check if there is a connection TO the point where you are standing.

6.8.3 Jump connections

Jump connections are a bit special as they cannot be drawn like any other connection. But that's not all, apart from that, Jump connections can also be one-way or two-way connections. To make matters even more complex, their 2-way version can come in two flavours: A "pure" two-way jump connection, i.e. a Jump connection from A to B and another Jump connection back from B to A or a "mixed" two way connection, with a Jump connection leading from A to B and a regular one-way connection back from B to A. The latter version will be very rare, though. Now once again, this sounds more complicted than it actually is. The Jump connection horizontal line displayed in red(Outgoing Jump connection), viewed from the waypoints where the bots will start their jump can be seen in the picture below.

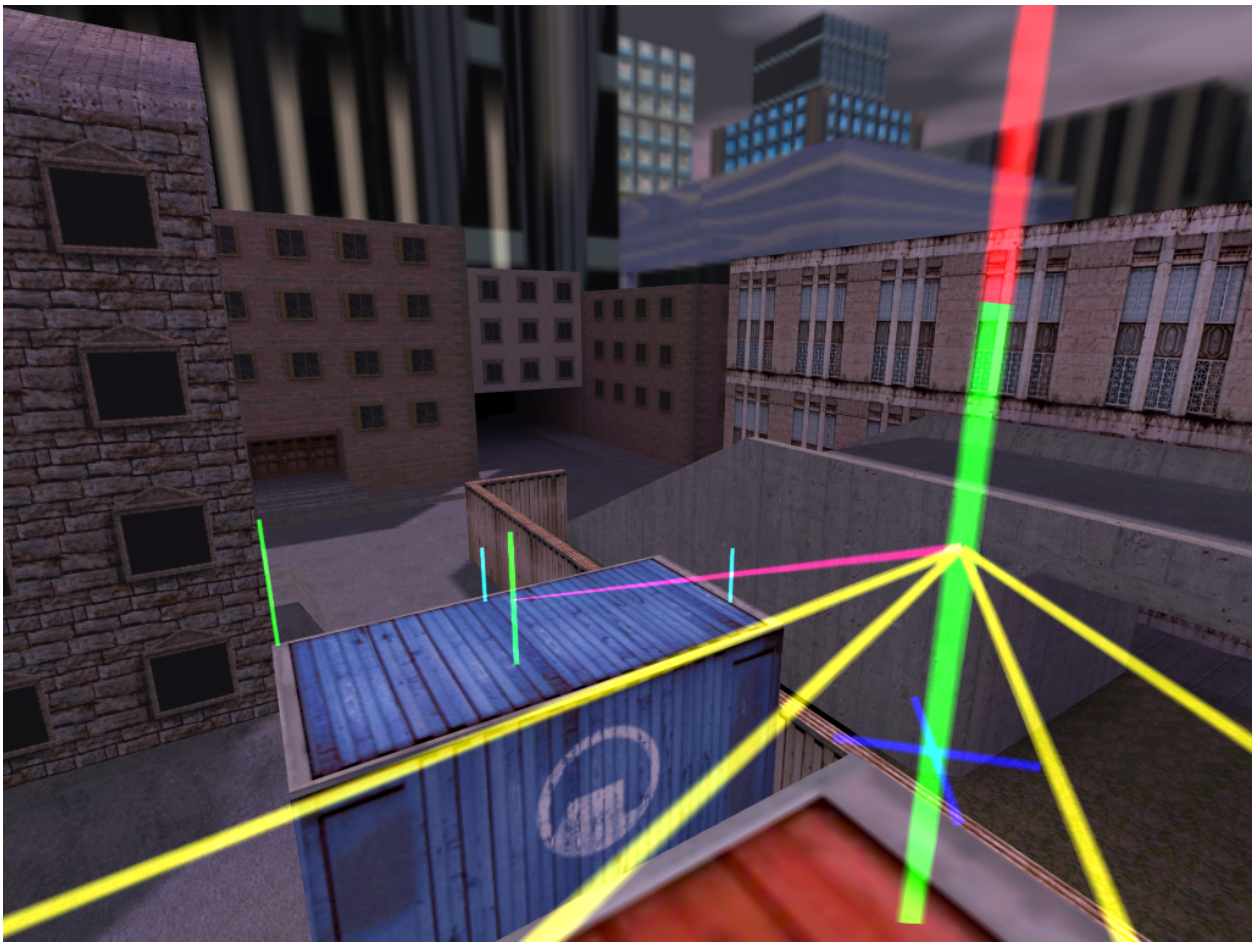

Fig. 21: An Outgoing Jump connection

6.9 Adding/Removing connections manually

Now you know the most important things about connections in general and the different types of connections. You also know how to adjust automatic connection lengths and how to add a jump connection by hand. But how do you add or remove a connection by hand? You simply aim at the desired waypoint and select the action to perform from the on-screen menu!

Here's how it works (Look at the screenshot below), let's pretend we wanted to delete the connection from the waypoint where we're standing to the left waypoint near the wall, aim at the waypoint with your crosshair. As soon as the waypoint is selected, it will become bigger, and a little arrows will appear in front of it.

Note: This only works if you are standing near a waypoint and aiming at another one! If you are standing in a waypointless area, you won't be able to use this function because it needs two selected waypoints (The one where you are standing and the one you are aiming at).

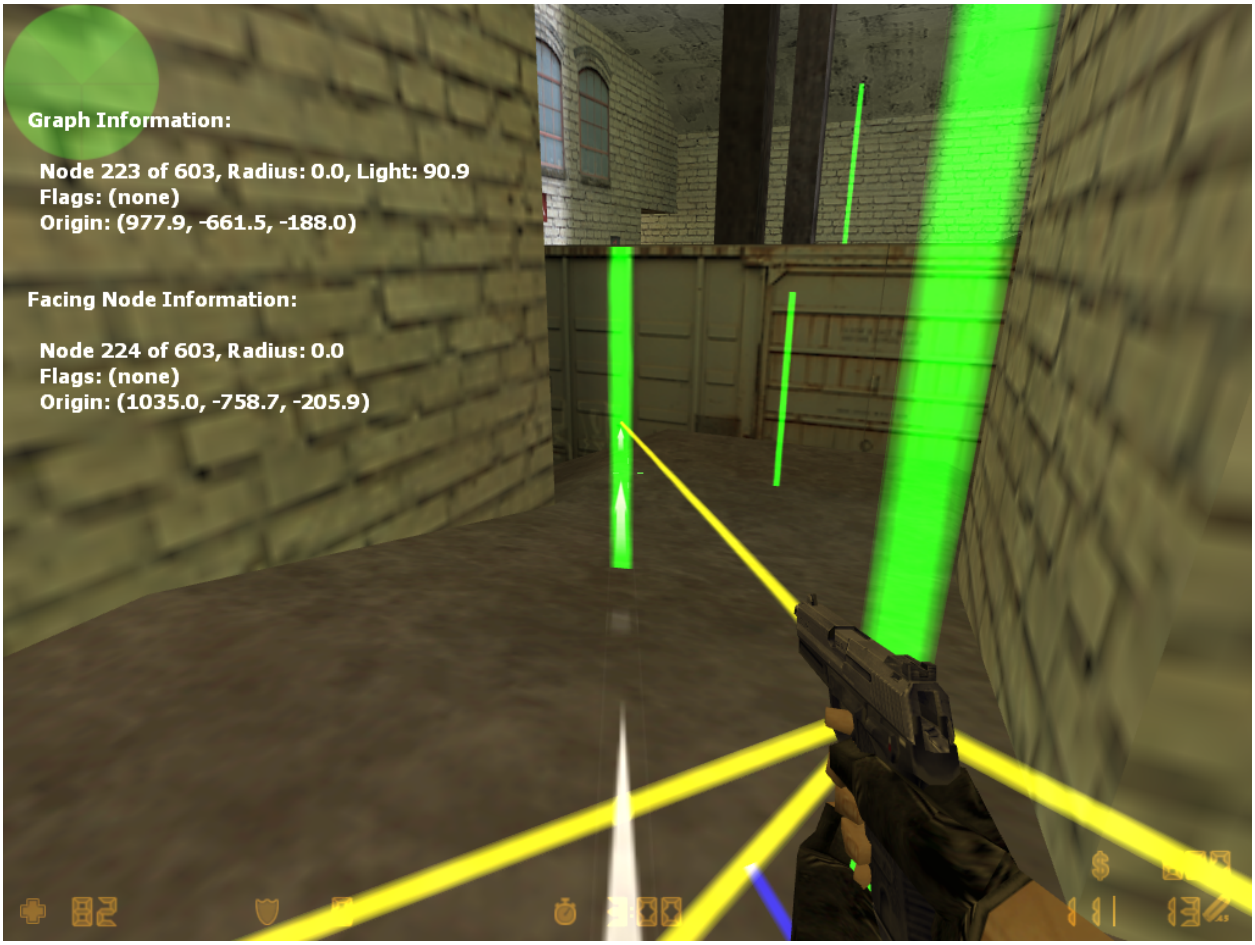

Fig. 22: The left selected waypoint

6.9.1 Removing path connections

To delete a path connection, you must open the waypoint menu and select `4. Delete path`.

After deleting the path connection, you may notice that the outgoing path connection has been removed (from the waypoint where you are standing to the waypoint you selected) as you can see in picture below:

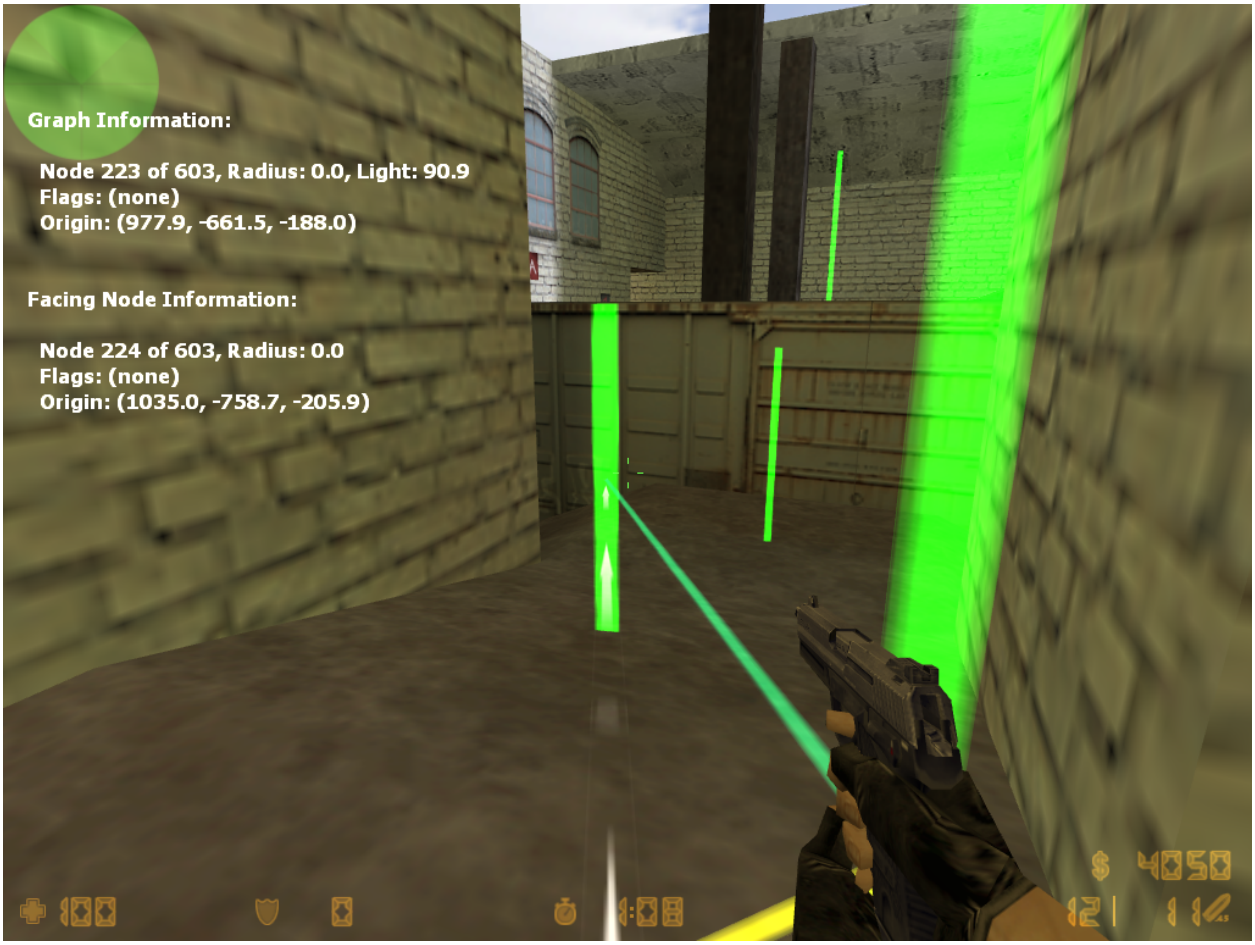

Fig. 23: Removed outgoing path connection, only incoming path connection remains (from the selected waypoint to the waypoint you are standing on)

Select `4. Delete path` again to delete the incoming path connection.

6.9.2 Adding path connections

To add a path connection, you must open the waypoint menu and select `3. Create path`. Then a menu should appear as shown in the picture below.

1. Select `1. Outgoing Path` to create an outgoing path connection from nearest to faced (or cached) node.
2. Select `2. Incoming Path` to create an incoming path connection from faced (or cached) to nearest node.
3. Select `3. Bidirectional (Both Ways)` to create a bi-directional (2-way) path connection between the nearest and faced (or cached) node.
4. Select `4. Jumping Path` to create an outgoing jumping path from nearest to faced (or cached) node.

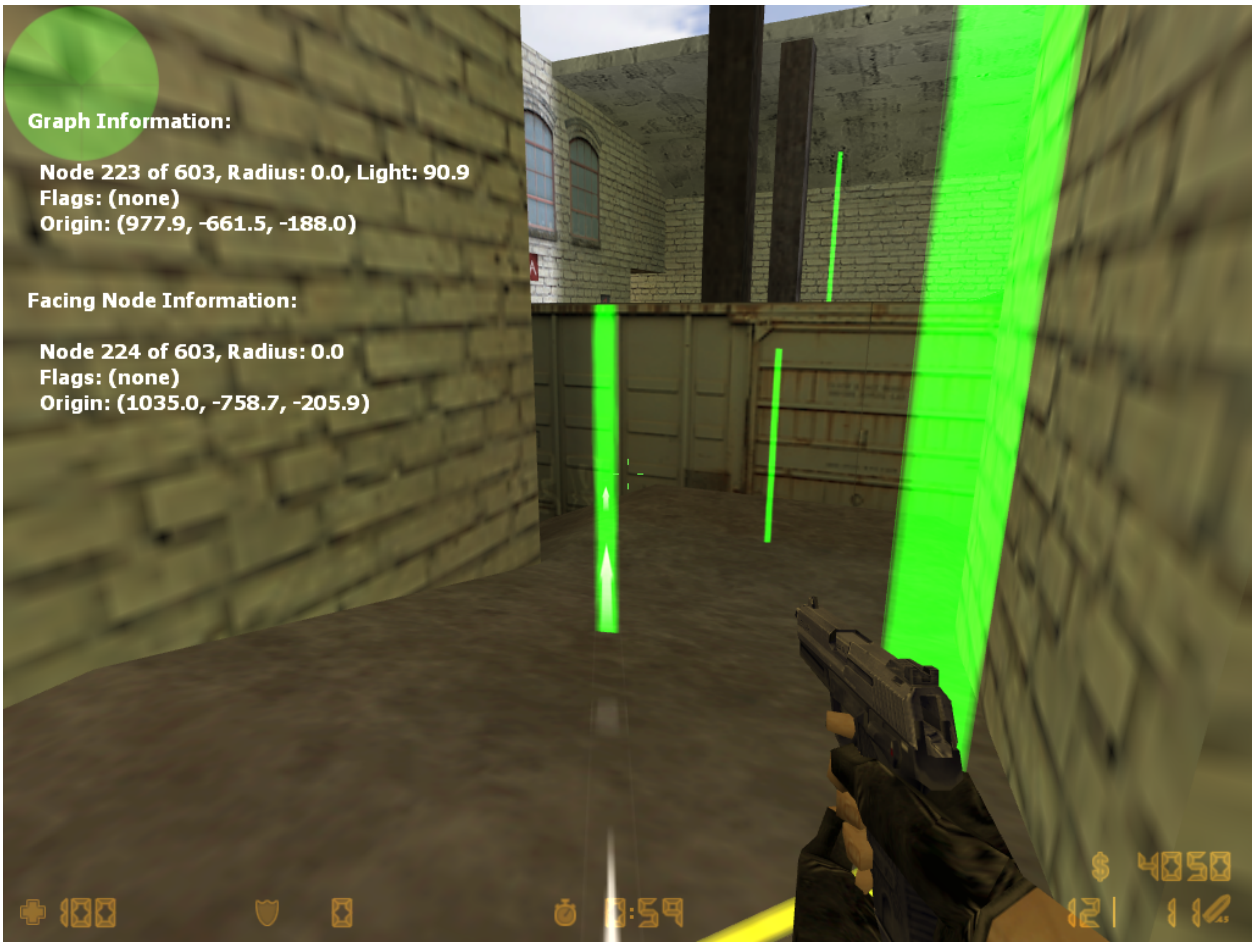

Fig. 24: Fully deleted path connection

As you have already noticed, all path connections have been removed from the waypoint you are standing on to the selected waypoint.

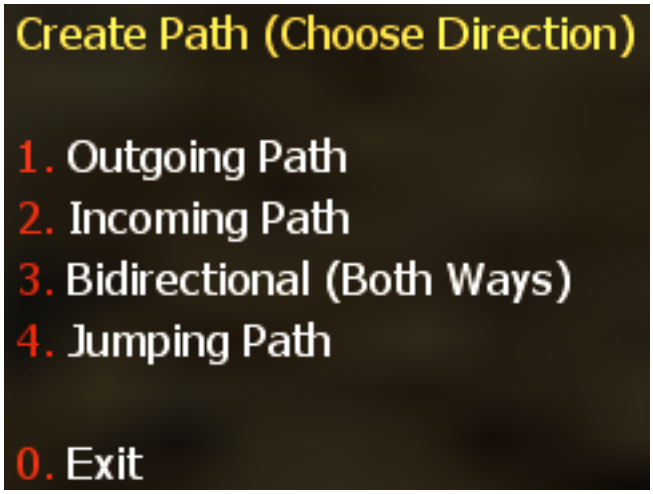

Fig. 25: Path directions menu

6.10 Waypoint Flags

YaPB has 9 flags for waypoints, which are:

1. Block with Hostage - a flag that prohibits Counter-Terrorists leading hostages to go along certain waypoints marked with these flags. Important: you should definitely put these flags on the ways where counter-terrorists can lose hostages!
2. Terrorist Specific - makes waypoint important for Terrorists
3. CTs Specific - makes waypoint important for Counter-Terrorists
4. Use Elevator - a flag for a waypoint that makes bots wait while they uplift on the elevator (you must put this flag on the waypoint at the beginning and at the end of the elevator path)
5. Sniper Point - a flag that makes the camp point as a sniper point (bots will only camp with sniper rifles).
6. Map Goal - a flag that turns an normal waypoint into a goal waypoint.
7. Rescue Zone - a flag that specifies the waypoint as a hostage rescue point.
8. Crouch Down - a flag that causes bots to crouch when reaching this waypoint.
9. Camp Point - a flag that makes the waypoint as a camp point. If you add this flag, it opens the menu to choose the start and end direction of the bot's view when it's camping.

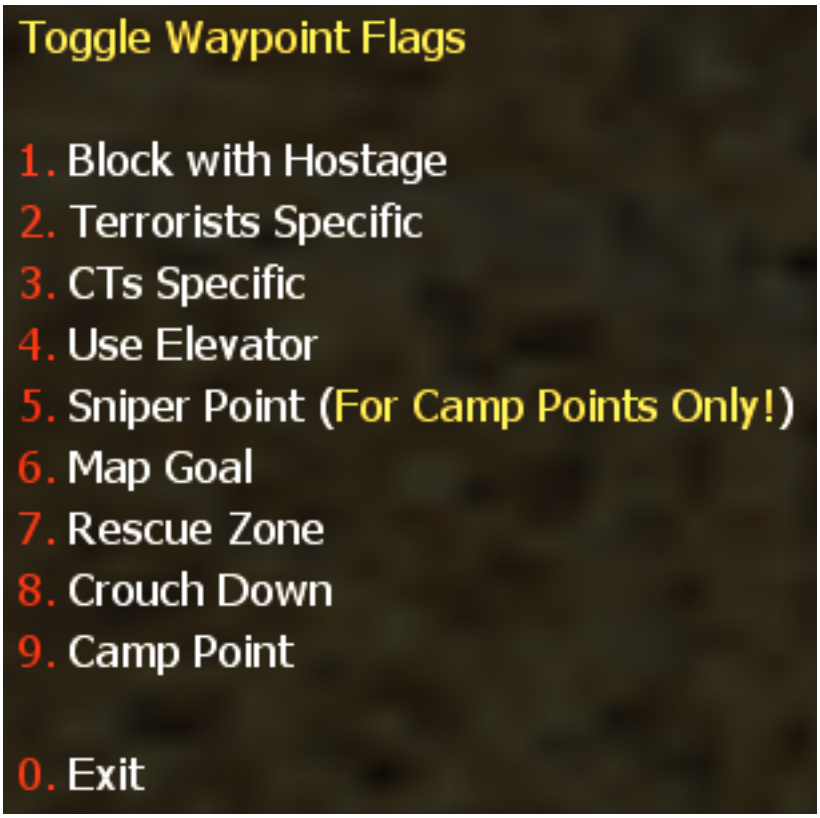

Fig. 26: Waypoint Flags Menu

6.11 Debug Goal menu

To test the bots walkability to a specified node, you need to open the second page of "Waypoint Operations" menu, and select `1. Debug Goal`. Then a menu should appear as shown in the picture below.

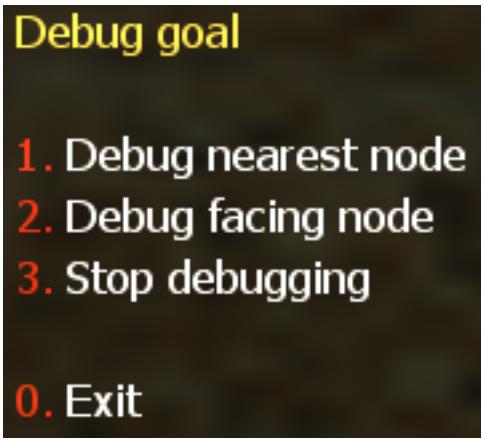

Fig. 27: Debug Goal Menu

1. Select `1. Debug nearest node` to specify the nearest node as the goal that the bot needs to reach.
2. Select `2. Debug facing node` to specify the facing node (which you pointed with your crosshair) as the goal that the bot needs to reach.
3. Select `3. Stop debugging` to disable the Debug Goal functionality.

CHAPTER

SEVEN

LOCALIZATION

YaPB localization files are located inside `addons/yapb/conf/lang` folder. Main localization are located inside `**_lang.cfg` where `**` is two-letter language code For example: `ru` for Russian language, `de` for Deutsch language, etc. At the moment, YaPB has three languages: English, Russian and Deutsch

Also, in addition to localizing the main things of the bot, such as menus, pop-up messages, text in the console, etc. You can create a chat base and a list of nicknames in your language. `**_chat.cfg` for chat and `**_names.cfg` for nicknames respectively.

7.1 Main Localization

To translate the bot into your language, you need to create a file `**_lang.cfg` Then open the file and insert the original bot strings after the [ORIGINAL] line. Also insert translated strings after [TRANSLATED] line.

It should look like this:

```
[ORIGINAL]
Here are the original strings of the bot.

[TRANSLATED]
Here are the translated strings of the bot.
```

You can use for example ru_lang.cfg as a template for translation.

7.2 Chat Localization

You just need to create a file `**_chat.cfg` for bot chat in your language. You can use the file en_chat.cfg as an example.

See also:

See the *Customization* section for details.

7.3 Nickname Localization

You just need to create a file `**_names.cfg` for bot nicknames in your language. You can use the file en_names.cfg as an example

CHAPTER

EIGHT

BUILDING THE BOT

Table of Contents

8.1 Before you begin

If you have all the required packages, you can skip this section.

Note: You need to build 32-bit library, since Valve has dropped support for 64-bit HLDS.

8.1.1 Installing the Windows Packages

- Install the latest version of Visual Studio from https://visualstudio.microsoft.com/
- Launch the Visual Studio Installer, select the required Visual Studio edition and install all necessary components for C/C++ development.
- Install the latest version of Git for Windows from https://git-scm.com/download/windows/
- Run the installer, and follow the installation instructions.
- Install the latest version of Python from https://www.python.org/downloads/windows/
- Run the installer, and follow the installation instructions.
- Install the Meson and Ninja using pip, by entering the following command in your cmd or powershell window `pip install meson ninja`.
- You're done!

8.1.2 Installing the Linux Packages

- Install the latest version of GCC or Clang, by entering the following command in your terminal window `sudo apt install gcc`, or `sudo apt install clang`.
- Install the latest version of Git, by entering the following command in your terminal window `sudo apt install git`.
- Install the latest version of Python, by entering the following command in your terminal window `sudo apt install python3`.
- Install the gcc-multilib and g++-multilib package, by entering the following command in your terminal window `sudo apt install gcc-multilib g++-multilib`.
- Install the Meson and Ninja using pip, by entering the following command in your terminal window `pip install meson ninja`.
- You're done!

8.2 Building on Windows

- Clone the YaPB repository, by entering the following command in your Visual Studio Developer PowerShell or Command Prompt window `git clone --recursive https://github.com/yapb/yapb`
- Enter to the YaPB project directory: `cd yapb`
- Configure this project using Meson: `meson setup build`
- Compile the DLL library: `meson compile -C build`
- You're done! The compiled library is located at `build/yapb.dll`

8.3 Building on Linux

- Clone the YaPB repository, by entering the following command in your terminal window `git clone --recursive https://github.com/yapb/yapb`
- Enter to the YaPB project directory: `cd yapb`
- Configure this project using Meson: `meson setup build`
- Compile the .so library: `meson compile -C build`
- You're done! The compiled library is located at `build/yapb.so`

CHAPTER

NINE

CREDITS

9.1 Development Team

- jeefo - Current maintainer, and 2.x - 4.x series developer.
- $_Vladislav - Documentation, waypoints.
- Whistler - Original author of YaPB fork. (retired)

9.2 Special thanks

- Botman, creator of the first bot. He figured out how to put bots into CS.
- CountFloyd, creator of PODBot, on which this bot is created.
- KWo, current PODBot MM maintainer.
- Pierre-Marie Baty, Bots-United co-founder.
- Splorygon, original PODBot MM maintainer.
- evilspy, JKbotti author.
- The Storm, e[POD]bot maintainer.
- Immortal_BLG, author of YaPB3.
- Valve, for creating such a great game.
- SpArK, helped a lot on early versions.
- Overitab, a lot of waypoints.
- [PRince4], a lot of waypoints commits.

9.3 Waypointers & Testers

- Waypointers:
  - $_Vladislav
  - Overitab
  - headshot26
  - Ek

  - CoCoNUT
  - MarD
  - Bots-United Members
  - Reallite Labs
- Testers:
  - ZmifF
  - RR99i
  - FFreJuSE
  - TeMP
  - SaTaN
  - SpArK
- Others:
  - Ancient, for hosting yapb for some time.

9.4 Additional

There are much more people who helped to bring this bot alive, if you are not mentioned here, please notify.