



# GUIDE BOT FOR T3D PROGRAMMERS GUIDE

## OVERVIEW

*Guide Bot* is a universal all-in-one middleware solution for development of AI in modern 3D games. With the help of *Guide Bot* you can create AI entities with very big range of abilities: from fantastic monsters to modern time soldiers.

*Guide Bot* is very flexible and sophisticated solution and its is a very big piece of code. Nevertheless based on most advanced and complicated algorithms *Guide Bot* can be used by programmers of very different skills: from hobbyist beginners to full-time professionals.

If you're unfamiliar with modern games AI conceptions *Guide Bot* can be a good starting point.

If you game software professional and want do expand your *T3D* by adding enhanced AI functionality *Guide Bot* will be also smart choice.

Current version of documentation is incomplete, so if you have any question please use THIS FORUM.

## TUTORIALS

The best way to learn how to use *Guide Bot* is to examine scripts of *Gi-Bot soldier* (see this doc below). But in the future releases of *Guide Bot* there will be "how-to" following tutorials:

*BEGGINNER LEVEL TUTORIAL*

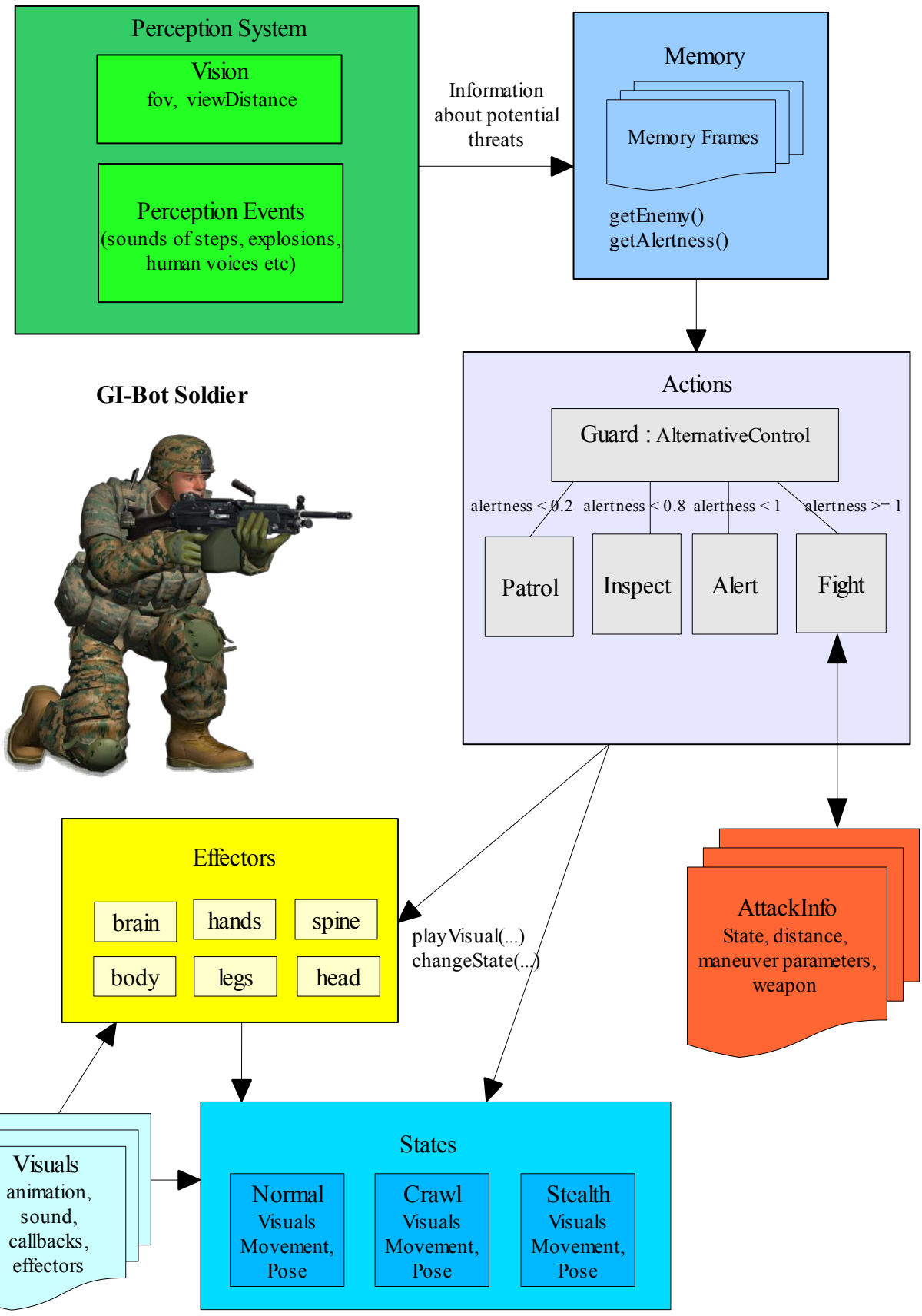
- Adding new bot to the world;
- Tweaking parameters of bots;
- Assigning teams (enemies, neutrals, allies);
- Adding perception events and configuring stealth behavior;

*INTERMIDIATE LEVEL TUTORIAL*

- Creating action from scripts;
- Declaring new states;
- Adding and editing new animation and sound as *visuals*;
- Creating and configuring new *AttackInfos*
- Tweaking movement and maneuvering.

*ADVANCED LEVEL TUTORIAL*

- Adding new actions in C++;
- Enhanced use of effectors;
- Linking actions with effectors.



## GI-BOT SOLDIER OVERVIEW

*Gi-Bot soldier* is based on *GuideBot* functionality and can be used as a starting point for learning main principles of AI architecture and algorithms behind *GuideBot.*

*Gi-Bot* based on *EnhancedPlayer* class – an enhanced version of original T3D's Player class, though all parameters of PlayerData datablocks are inherited and some new are added aswell. (*file: giSoldierPlayer.cs*)

*Gi-Bot* can do following:

- Guard some territory using predefined path of waypoints (ActionGuard, ActionPatrol*, file: actionGuard.cs*);
- When bot hears some suspicion sounds it will look around to see the source of the sound; when see the enemy bot will abandon patrolling and start fighting the enemy; (ActionFight ); when fighting bot will use covers (ActionCover), avoid grenades and exposing barrels (ActionAvoidDanger) (all actions in files: *actionFight.h, actionFight.cpp*)
- attack enemy with different types of attack (file: *giSoldierAttackInfos.cs*);

## STATES AND VISUALS

Visual and states of *Gi-Bot* described in *file: giSoldierStates.cs*

Visual is combination of animation, sound and script callbacks encapsulated in one datablock.

```
datablock VisualDataBlock(forwardVisual)
{       visualAnim = "run";
        visualSound = runSfxProfile;
        visualEffectors = "legs";
        callbackTime[0] = 0.327;  callbackFunction[0] = stepCallback;
        callbackTime[1] = 0.8;   callbackFunction[1] = stepCallback;
};
```

*visualEffector* is a list of effectors, visual will be assigned to. For example *"run"* animation should be on the *"legs"* effector, shoot

*callbackFunction* is a script function that will be called once *callbackTime* reached by animation.

*callbackTime* is set relatively to animation time, and must be in range of 0 (beginning of animation) to 1 (end of animation).

## ATTACK INFOS

Attack infos describe what types of attack can be used by bot during the fight (*file: giSoldierAttackInfos.cs*)

```
datablock AttackInfoDataBlock(SimpleAttack)
{
	minDist = 0;   // min distance to enemy for attack execution
	maxDist= 30;  // max distance to enemy for attack execution
	fov        = 20; // max fov of enemy for attack execution (max angle to enemy)
	periodicity = 0;    // periodicity of this attack (attack won't execute once  in period)
	evaluator   = "BaseWeaponEvaluator"; // attached function that returns numeric estimation
	manoeuvreMinDist = 15;  // maneuverer description: minimum distance (default: 2)
	manoeuvreMaxDist = 30; // maneuverer description: minimum distance (default: 2)
	strafe                  = true;    // maneuverer description: enable strafe movements(default:true)
	pause                   = 1500;  // pause between attack action (i.e."shoots")
	blockMovements = false;  // block movements during attack action(acquiring E_BRAIN_MOVING)
	action      = "ActionShot";  // background state
	state       = "StateNormal"; // background state
	weapon      = "MachineGunImage";
	ammo        = "BlasterAmmo";
	coverState          = "";
	attackState         = "";      // attack action state (i.e. special state for "shoots")
	lookAtEnemy = true;        // turn body and head to the enemy during attack (default:true)
};
```