Game Architecture

Artificial Intelligence

Today’s Agenda

- What are we doing again?
- Character control
- Navigation and other basic actions
- Higher level actions and decision making
- Game designer control
- Terminus AI retrospective

What are we doing?

- AI as method of non-player character control
- AI serves game design
  - AI realistically models human behavior if game design calls for it
  - AI challenges the player if game design calls for it
  - AI is objectively awful if the game design calls for it
  - **The goal is always an excellent game**
- State-of-the-art game AI is complex, broad, and deep
  - Much more than we can hope to cover in this lecture
  - Take heart; players have difficulty inferring AI intent, or even, relative intelligence
  - **The only right answers are those that enable excellent gameplay**

Character Control

Typically:

- A finite state machine that describes how character is allowed to behave
  - E.g. When standing, transition to jumping when jump action is taken
- FSM expresses logical character state
  - Animation may have its own state machine
- FSM is shared with player characters and non-player characters

Navigation

Agents traversing a game world:

- Racing; accelerating, braking, shifting, following the track
- Third person shooter; running, jumping, ducking, avoiding obstacles
- 2D platformer; move left/right, jumping, falling

General approach:

1. Model the game world
2. Using model & current state, generate inputs for character state machine

Navigation: Modeling the World

- Racing game model:
  - Track as 3 or more splines (bounding the drivable surface and indicating best path)
  - Encode data into control points, such as suggested speed, when to jump, etc.
- Third person shooter:
  - Game world a series of navmesh objects
    - Collection of connected convex polygons
    - Unobstructed straight line paths within polygons
    - Graph search (A* or similar) for multi-polygon paths
    - Edges of polygons may store additional data for traversal (jumping, etc)
- 2D platformer:
  - Graph connecting each platform
  - Graph search to path plan
  - Graph edges encode additional traversal information

Navigation: Driving the Character

Some options:

- Generate virtual button presses on a virtual controller
  - Makes it easy swap between playable and NPC
  - One system to rule them all
  - Virtual controller might not be expressive enough for exotic AI
- Generate the side effects of button presses directly
  - Still honest, but perhaps more flexible
- Divorce player control and NPC control in more places
  - Make player physics different than NPC physics
  - Probably not a good idea

Other basic actions

Anything that maps to a button press by the player.

For example:

- Firing a weapon
- Opening a door
- Etc.

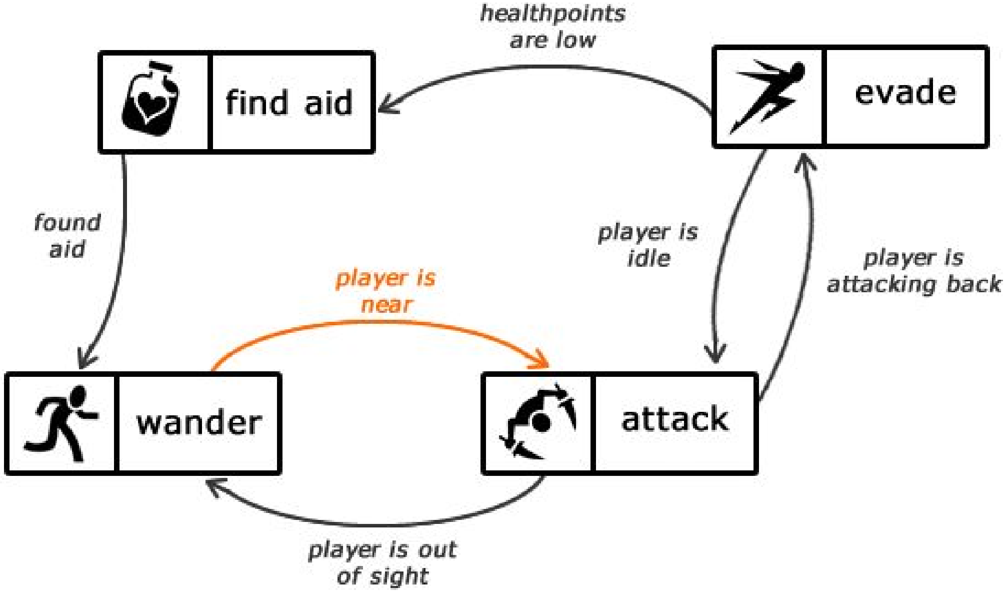
Higher Level Actions

String together basic actions to achieve goals.

- Most agents share the same set of basic actions
- Classes of agents distinguish themselves through higher level actions

Lots of different ways to organize high level actions.

High level action: FSM

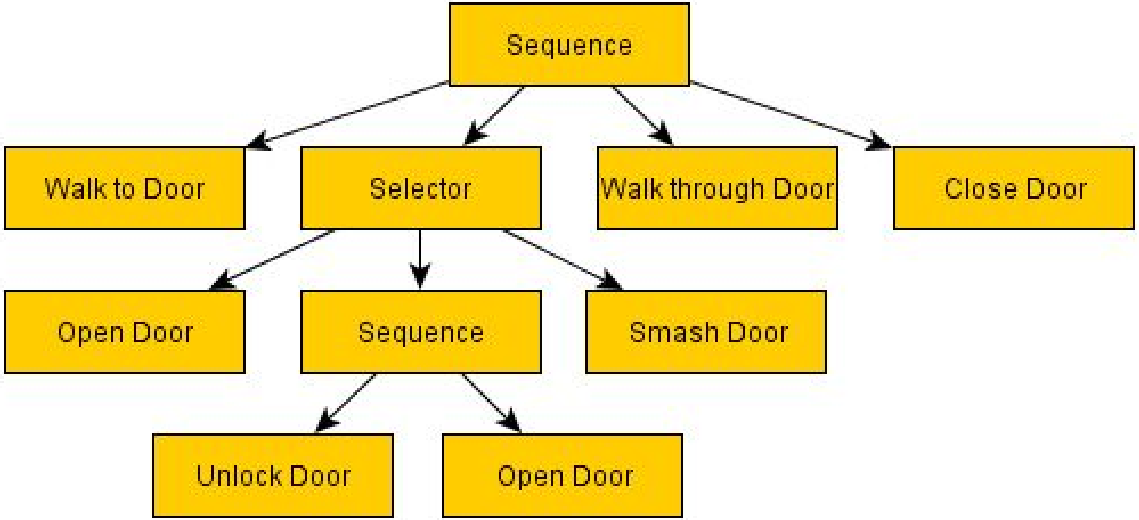


FSMs In Brief

Typically:

- Each state is implemented in one or more functions
    - Big switch statement maps state to function
- Transition logic is encoded in state functions
- Stack of active states
    - Top of stack is executed each frame
    - Can push new state and/or pop self to transition

High level action: Behavior Trees

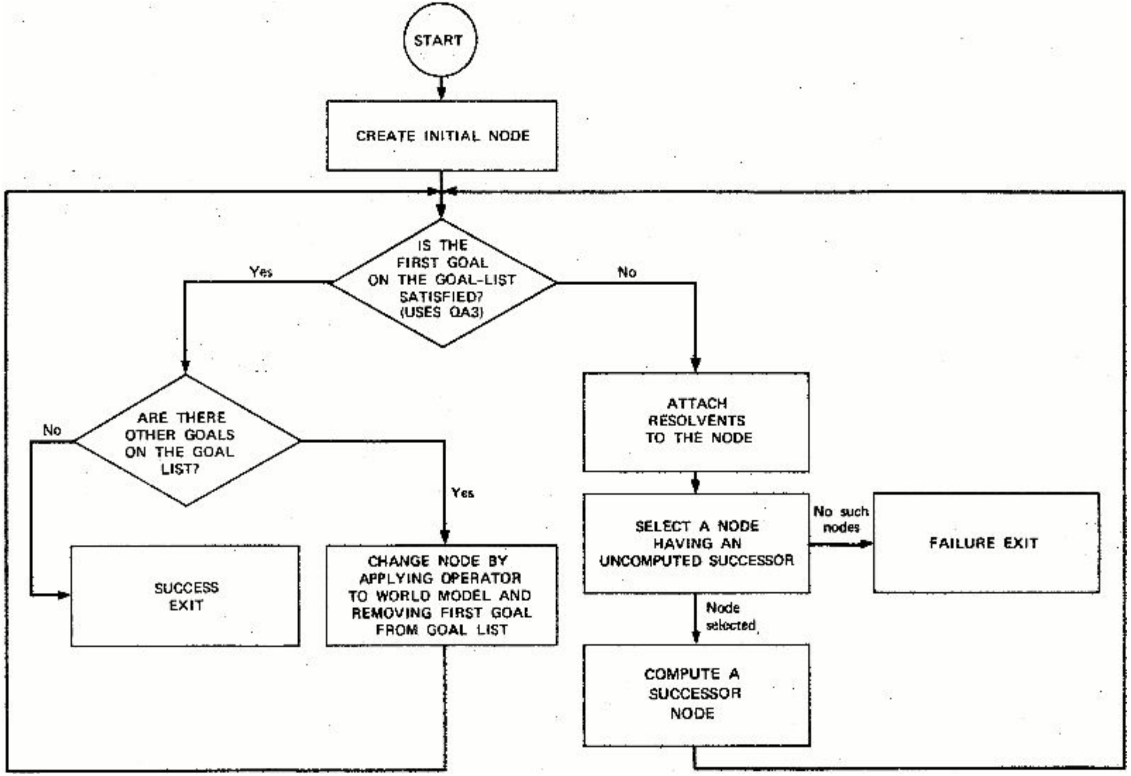


Behavior Trees In Brief

Typically:

- Structure
  - Leaf nodes are commands that control entity
  - Interior nodes select what to do
- Action
  - At some rate, evaluate the behavior tree and decide what to do
  - If what we want to do represents a change in active node:
    - Close old subtree (may take multiple frames)
    - Open new subtree (may take multiple frames)
  - Updating/ticking a node returns a result
    - Success, failure, running
  - Parent nodes act on child results

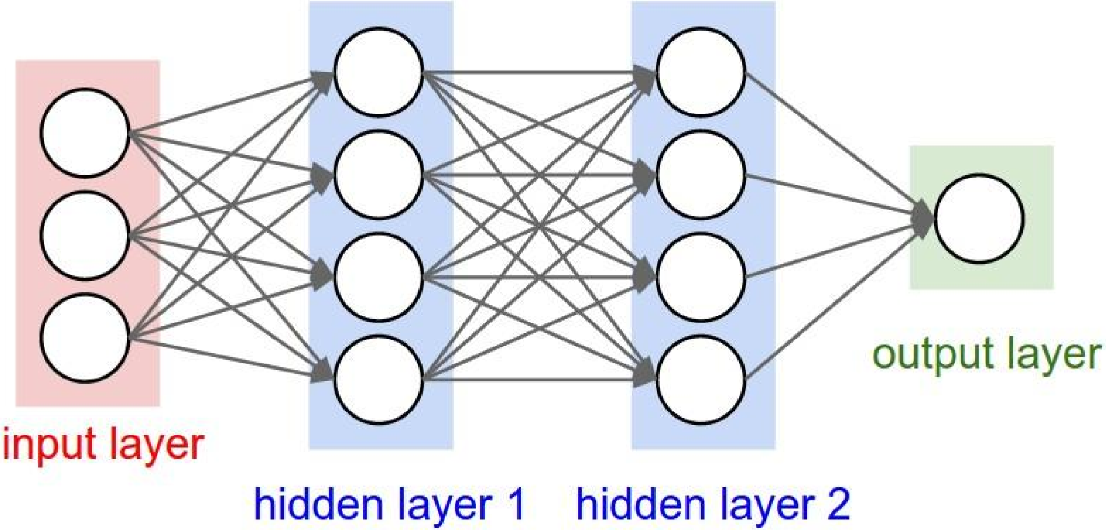
High level action: Planners


START
CREATE INITIAL NODE
IS THE FIRST GOAL ON THE GOAL-LIST SATISFIED? (USES QA3)
Yes
No
ARE THERE OTHER GOALS ON THE GOAL LIST?
No
Yes
SUCCESS EXIT
CHANGE NODE BY APPLYING OPERATOR TO WORLD MODEL AND REMOVING FIRST GOAL FROM GOAL LIST
ATTACH RESOLVENTS TO THE NODE
SELECT A NODE HAVING AN UNCOMPUTED SUCCESSOR
No such nodes
FAILURE EXIT
Node selected
COMPUTE A SUCCESSOR NODE

Planners In Brief

- More declarative, less imperative.
- Given *operators* with preconditions and side-effects, decide on a course of action to accomplish some goal.
    - Operator: `PickupWeapon`
        - Argument: `Weapon`
        - Precondition: `NearWeapon`
        - Side effect: `HaveWeapon`
- Planners (such as *STRIPS*) use search algorithms to find operators to accomplish goals (such as A*).

High level action: Neural Networks



Neural Networks In Brief

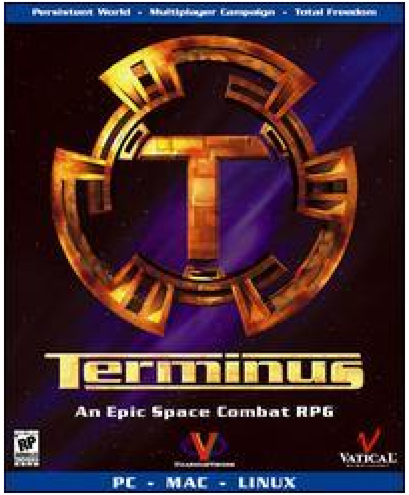
- In general:
  - Organized in layers
  - Problem is modeled in a small number of discrete inputs
  - State of inputs are applied
  - Signal propagates through network, each node applies a weight to the signal
  - Solution is given at output layer
- For gaming:
  - Input might be some near term goal and world state
  - Output might be virtual controller actions
- Neural networks must trained with some data
  - Ideally node weights converge such that certain types of input produce certain output
- Neural networks excel when:
  - There are patterns to be recognized
  - Potential dataset is large, not well understood, and conventional methods failed

Game Design Control

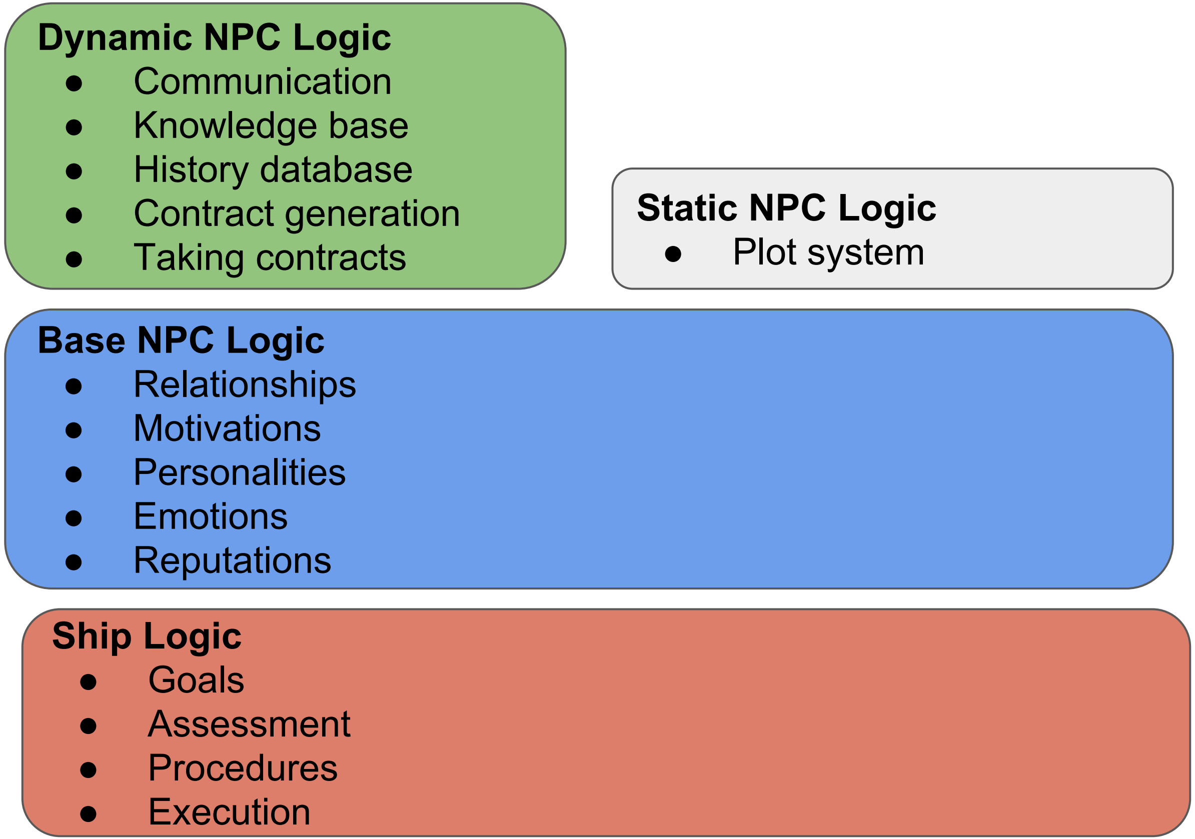
- AI is tightly coupled to game design
- Maybe:
  - Game design builds behavior trees
  - Game design uses script language to write behavior tree or FSM action nodes
  - Game design supplies AI with goals
- Interface for game designers to specify and control AI is important
  - How game designers should work with AI is key technical requirement

Case Study: AI in Terminus

- Multiplayer space combat RPG circa 2000
- Certain ‘realistic’ elements and setting:
  - Space flight with Newtonian physics
  - Fairly accurate model of the solar system
  - Fairly complex spacecraft model
  - Simulated economy facilitating trading
  - Main story ran over 24 hours, real-time
- Some things the AI should do:
  - Basic navigation, attack, defense
  - Outfit new craft, repair existing craft
  - Post contracts, collect bounties
  - Mine asteroids, trade goods
  - Further interests of major political actors
  - Run indefinitely in freeplay mode



Layers of Terminus AI



Dynamic NPC Logic

- Communication
- Knowledge base
- History database
- Contract generation
- Taking contracts

Static NPC Logic

- Plot system

Base NPC Logic

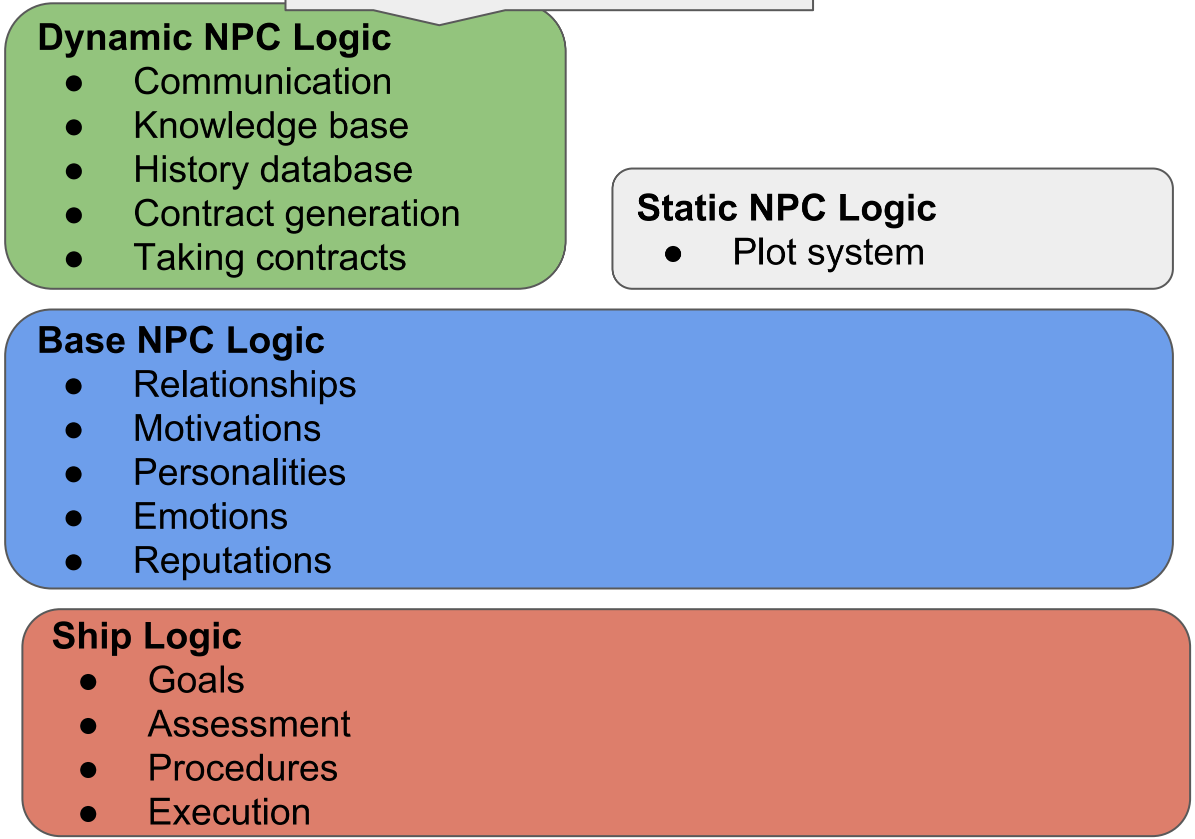
- Relationships
- Motivations
- Personalities
- Emotions
- Reputations

Ship Logic

- Goals
- Assessment
- Procedures
- Execution

Layers of Termin

‘Self determining’ AI actors.



Dynamic NPC Logic

- Communication
- Knowledge base
- History database
- Contract generation
- Taking contracts

Static NPC Logic

- Plot system

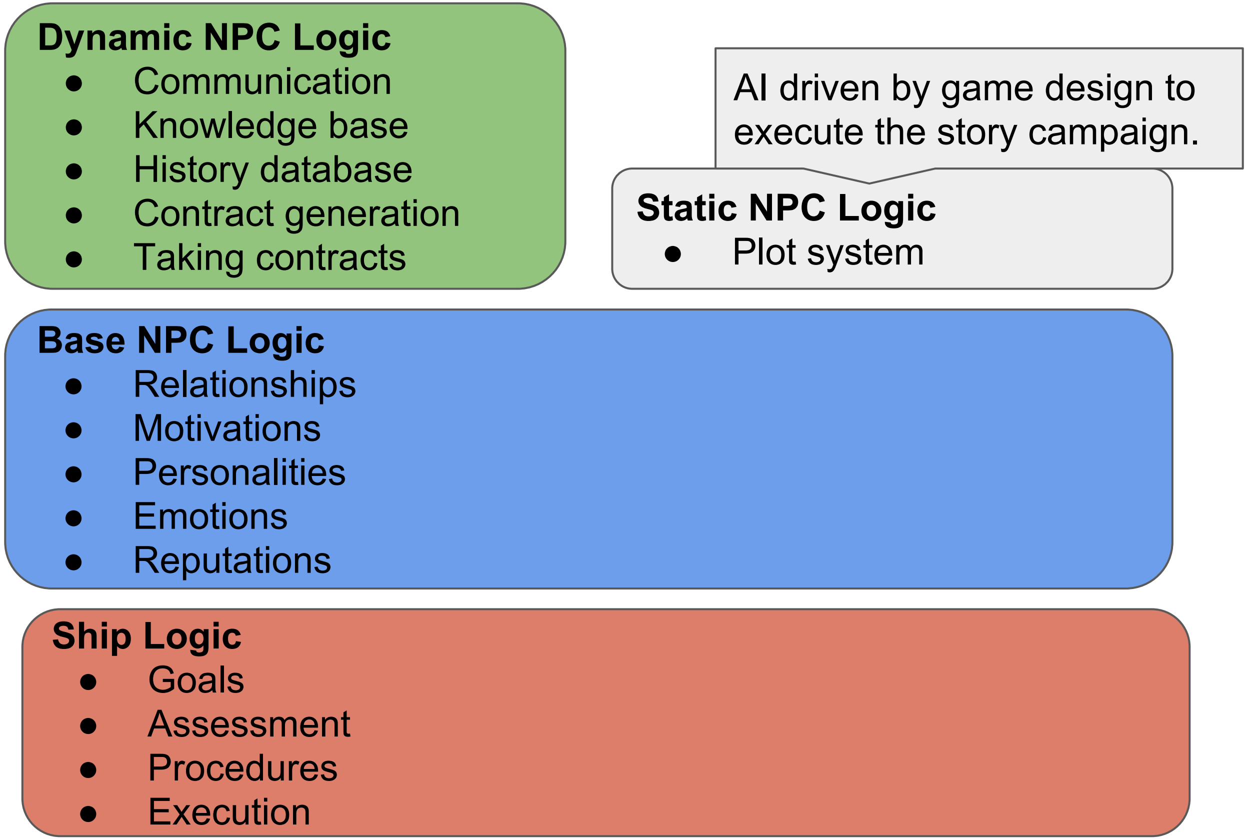
Base NPC Logic

- Relationships
- Motivations
- Personalities
- Emotions
- Reputations

Ship Logic

- Goals
- Assessment
- Procedures
- Execution

Layers of Terminus AI



Dynamic NPC Logic

- Communication
- Knowledge base
- History database
- Contract generation
- Taking contracts

AI driven by game design to execute the story campaign.

Static NPC Logic

- Plot system

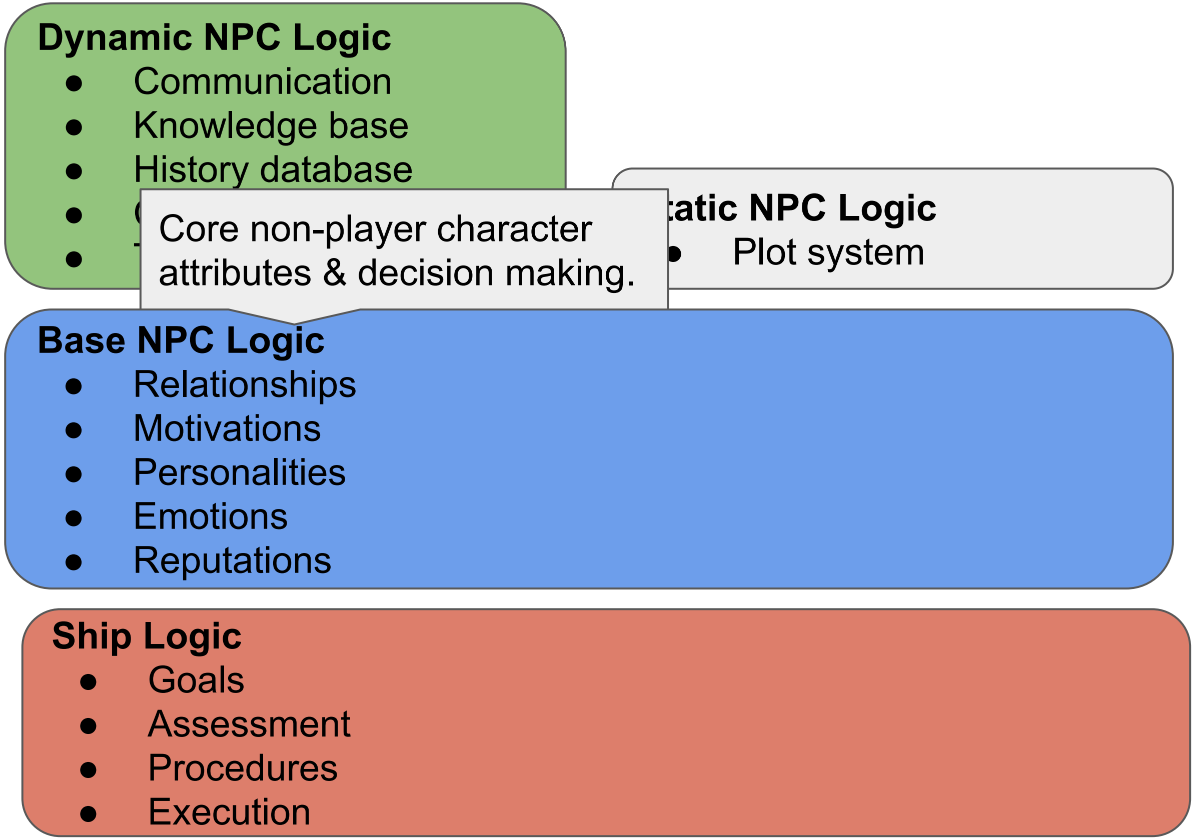
Base NPC Logic

- Relationships
- Motivations
- Personalities
- Emotions
- Reputations

Ship Logic

- Goals
- Assessment
- Procedures
- Execution

Layers of Terminus AI



Dynamic NPC Logic

- Communication
- Knowledge base
- History database
- C
- T

Core non-player character attributes & decision making.

tatic NPC Logic

- Plot system

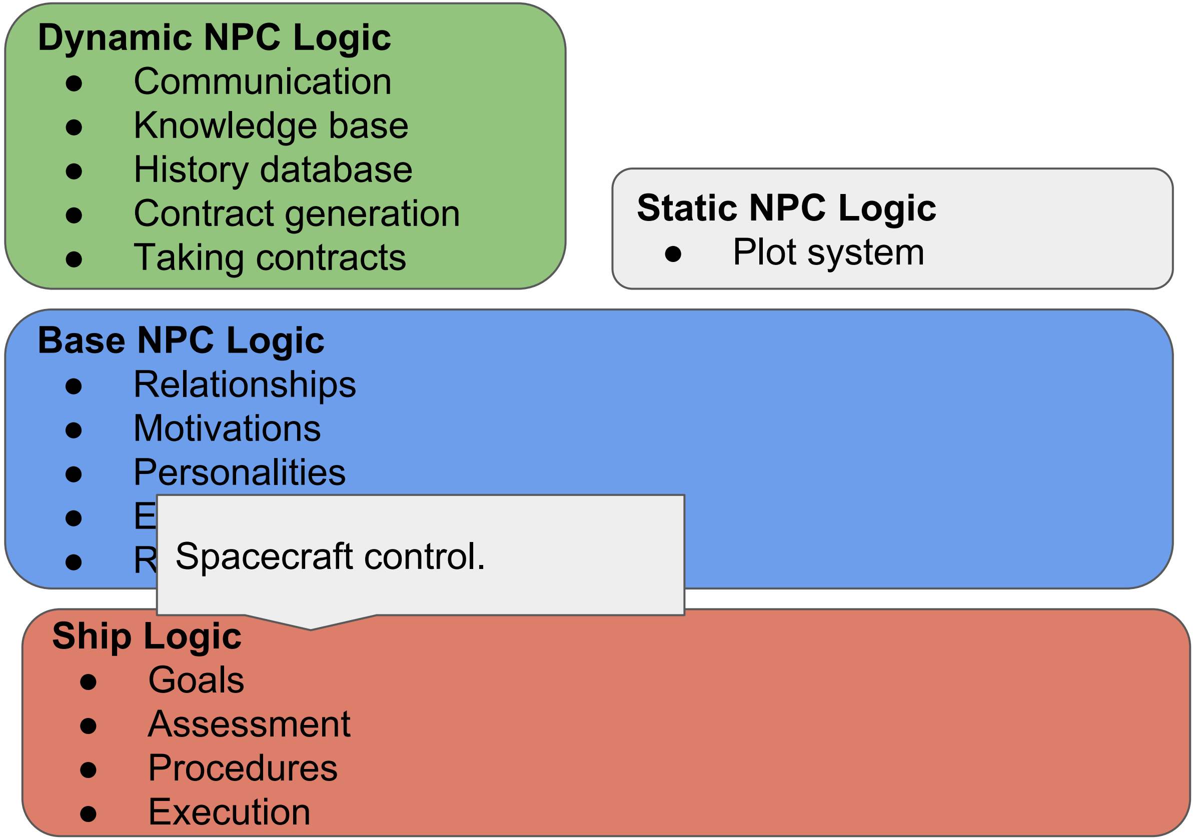
Base NPC Logic

- Relationships
- Motivations
- Personalities
- Emotions
- Reputations

Ship Logic

- Goals
- Assessment
- Procedures
- Execution

Layers of Terminus AI



Dynamic NPC Logic

- Communication
- Knowledge base
- History database
- Contract generation
- Taking contracts

Static NPC Logic

- Plot system

Base NPC Logic

- Relationships
- Motivations
- Personalities
- E
- R

Spacecraft control.

Ship Logic

- Goals
- Assessment
- Procedures
- Execution

Ship Logic

Stack of 4 layers:

1. Goal - High level spacecraft control.
   a. Travel to far away location, attack a target, patrol an area, hijack a ship, etc.
2. Assess - Situational awareness and action.
   a. If critically low on fuel, go to nearest space station.
   b. If high priority enemy unexpectedly near, attack.
   c. Else, proceed with goal.
3. Procedure - Mid-level spacecraft control.
   a. Docking, combat decision making.
   b. Raycasts and the Combat Decision Matrix.
4. Execute - Low-level spacecraft control.
   a. Collision avoidance, weapon fire, and thruster control.
   b. More raycasts, linear algebra, physics.

Ship Logic Part 2

Each frame:

- Commands passed from goal layer down to execution layer.
- Command is modified as it is passed down stack.
  - Situational assessment may choose to another command.
  - High level commands get broken down into low-level components.
- Ultimately a weapon or thruster probably fires.

Base NPC Logic

An array of factors that influence decision making:

- Relationship matrix defines affinity and respect between all NPCs.
- Motivations, personalities, emotions, reputations…

For example, player shoots near an NPC. NPC responds based on:

- Relationship with player
- Emotional state
- Motivational state
- Player reputation

Static NPCs

- Will respond to the situation if warranted
- Else, will follow goals as provided by plot system
  - Plot system = the story script as authored by game designers
  - Plot system provides goals to static NPCs (created themselves by the plot system)
  - Plot system was game designs primary interface to AI control

Dynamic NPCs

- Generate and take *contracts* (e.g. goals).
  - Dynamic NPCs are self determining.
- To do this, dynamic NPCs draw on several sources:
  - Knowledge base
    - Public or private knowledge of future events (transport runs, patrol routes, etc.)
  - History database
    - Recording of all other characters encountered.
  - Economy
    - Price of ~100 staples traded in the solar system.
  - Situational analysis
    - What’s visible on scanners, what’s in the news, etc.
  - Communication
    - When nearby friends, history and knowledge is shared.
  - Emotion, personality, etc.

Terminus AI Outcomes

- Interplay of scripted story and dynamic NPCs led to interesting and unexpected events. Frequently emergent behaviors.
- Dynamic NPCs were able to take the role of the player whenever the player was absent. Scaled easily in multiplayer story mode.
- 99% of what the AI was doing was invisible to most players.
- Did a generally poor job with player communication.
- Debugging was ‘interesting’.

Summary

AI is very wide subject. We just scratched the surface.

- Character control
- Navigation and other basic actions
- Higher level actions and decision making
- Terminus AI

End Lecture

Where to go for more…

https://github.com/recastnavigation/recastnavigation

https://gamedevelopment.tutsplus.com/tutorials/finite-state-machines-theory-and-implementation--gamedev-11867

http://aigamedev.com/open/article/strips-theorem-proving-problem-solving/

http://www.ai-junkie.com/misc/hannan/hannan.html