OPERAČNÍ PROGRAM PRAHA ADAPTABILITA

EVROPSKÁ UNIE

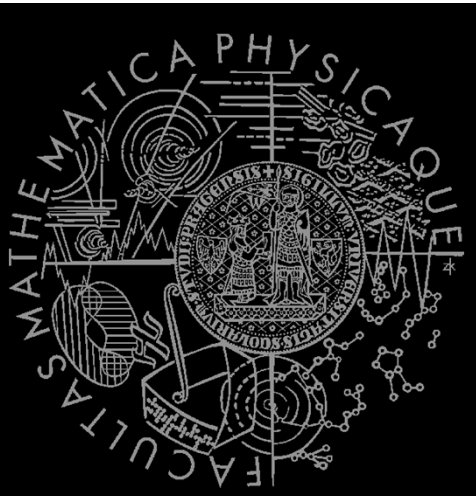
EVROPSKÝ SOCIÁLNÍ FOND

Pogamut 3
Lekce 5 - Navigace

PRAHA & EU

INVESTUJEME DO VAŠÍ BUDOUCNOSTI

Faculty of mathematics and physics
Charles University in Prague
28th March 2013



UT2004 bots made easy!

Pogamut 3

Lecture 5 – Navigation

Warm Up!

- Fill the short test for this lessons
  - 5 minutes limit
  - http://alturl.com/thkt7

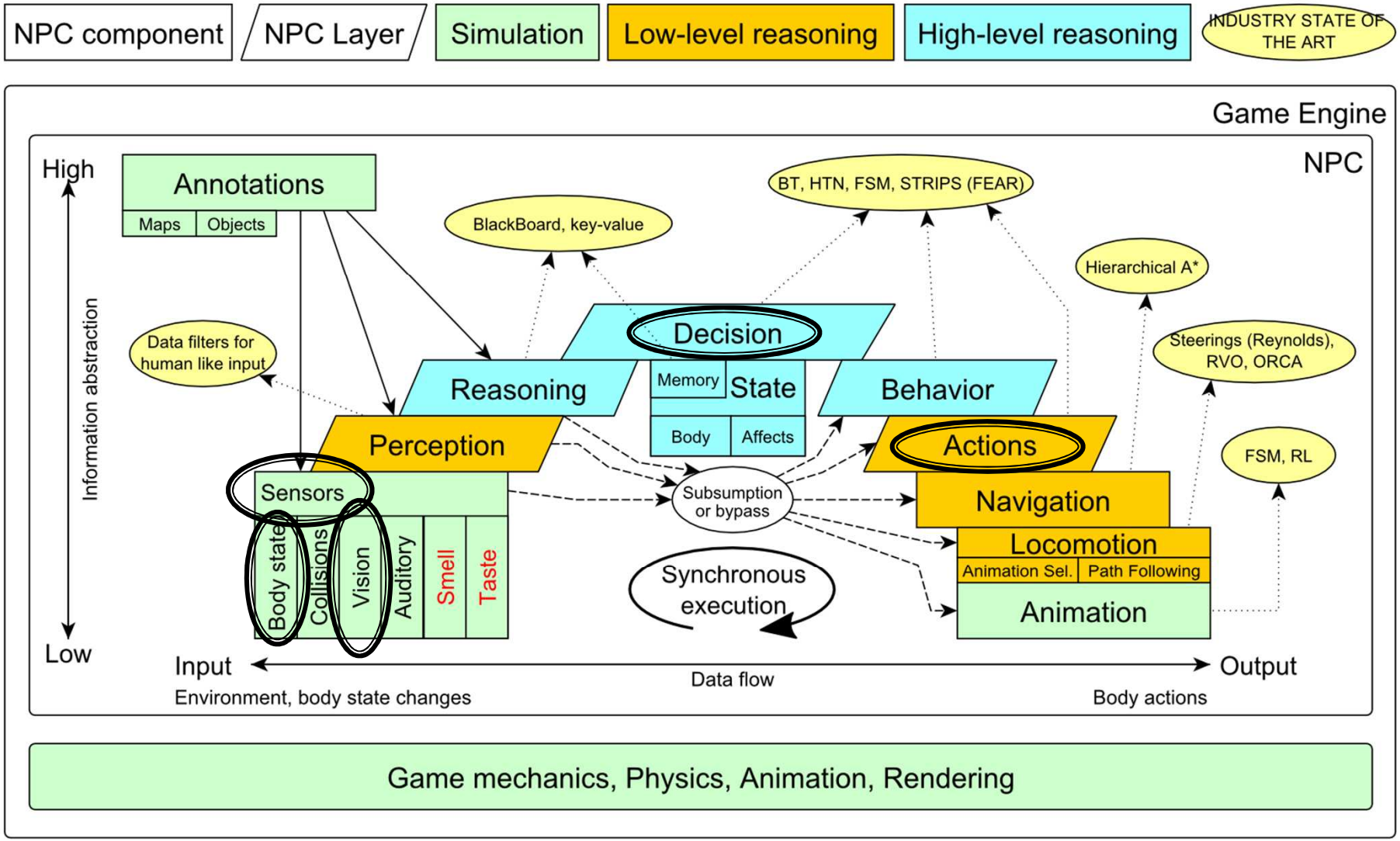
  - https://docs.google.com/forms/d/1cxRjtJoCVRH8D--_eQwK1UVyu5bCyqGVmH6VtNwvcVY/viewform

Today's menu

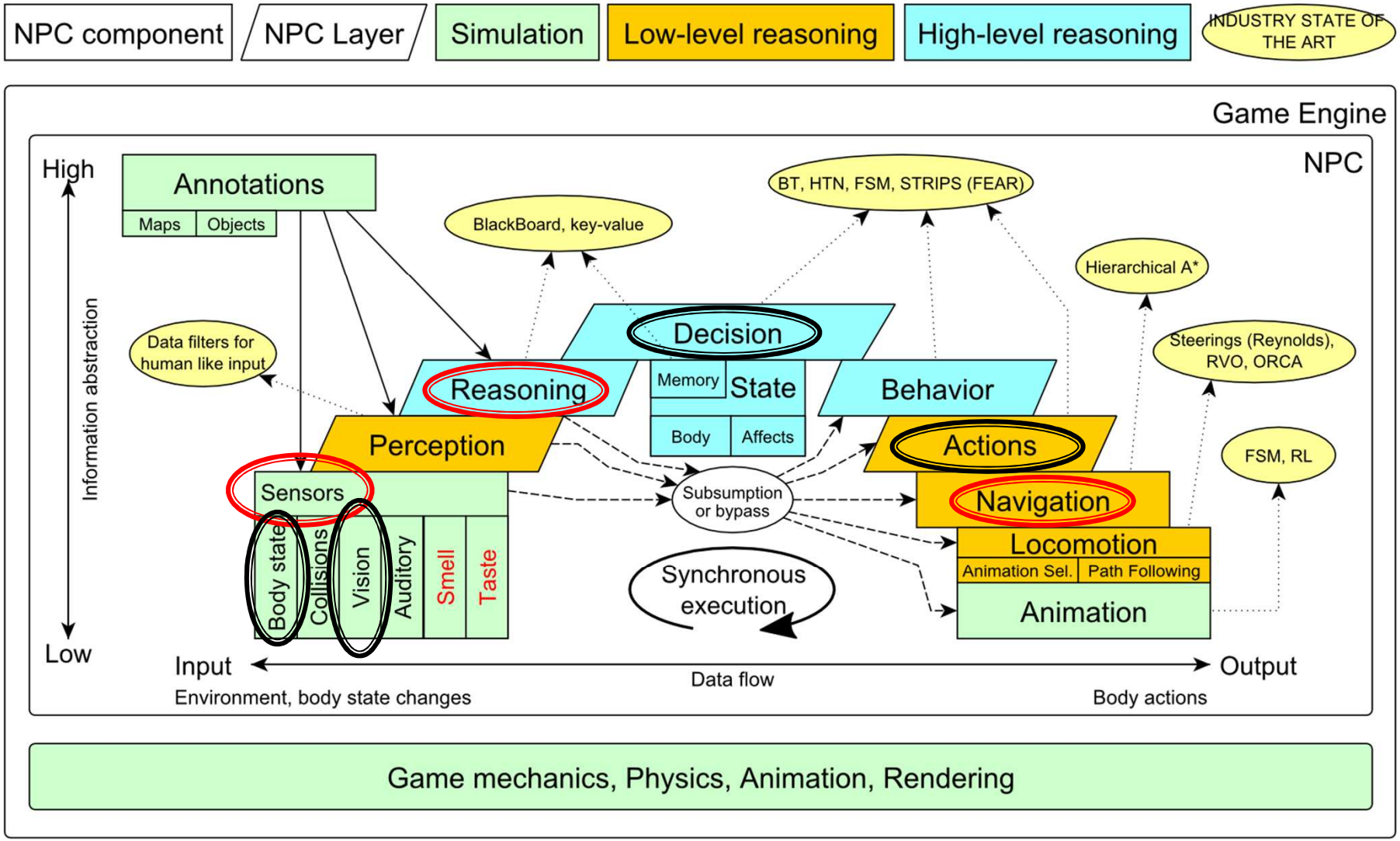
Navigating inside UT2004

1. **Big Picture**
2. PogamutUT2004 World Abstraction
3. PogamutUT2004 Navigation

Big Picture
Already covered



Big Picture
Today



Today's menu

Navigating inside UT2004

1. Big Picture
2. **PogamutUT2004 World Abstraction**
3. PogamutUT2004 Navigation

Pogamut World Abstraction
Basics

Objects (IWorldObject):

- Player
- Item
- NavPoint
- Self
- IncomingProjectile

Events (IWorldEvent):

- HearNoise & HearPickup
- BotDamaged & BotKilled
- PlayerDamaged & PlayerKilled,
- Bumped
- GlobalChat

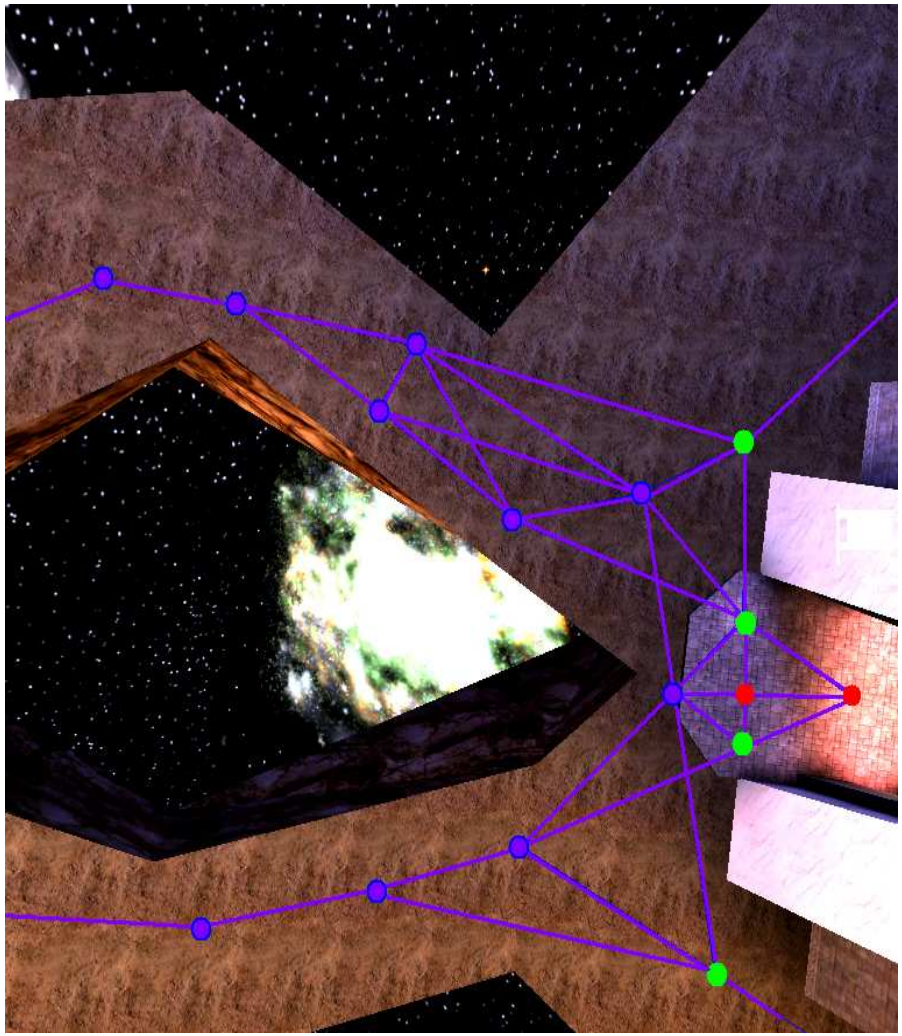
- Use modules, listeners and Pogamut helper classes!
  - this.players, this.items, this.info …
  - MyCollections, DistanceUtils

```
if (this.players.canSeePlayers()) {  …  }

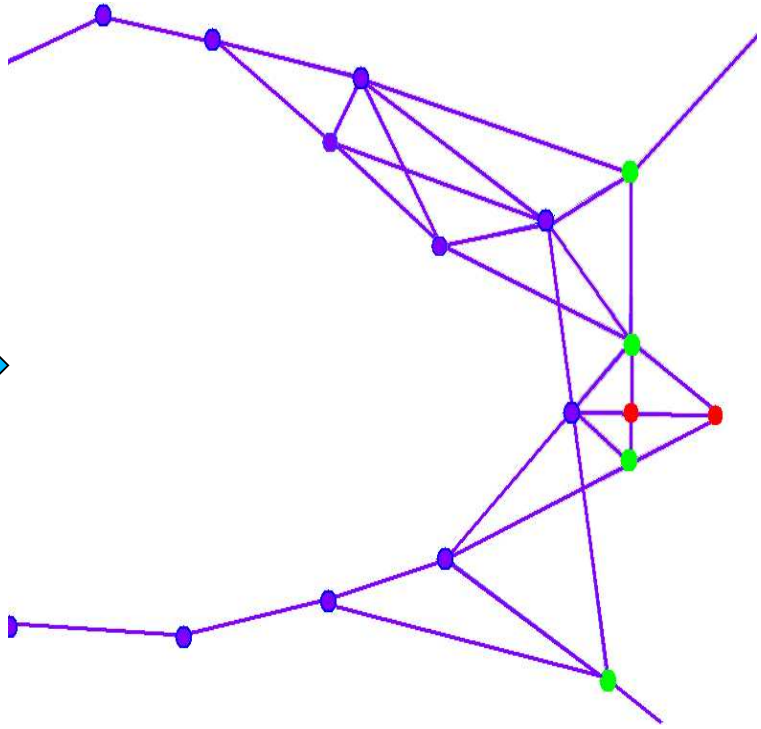
@EventListener(eventClass = GlobalChat.class)
public void chat(GlobalChat chatEvent) {
    …
}
```

UT2004 World Abstraction

Navigation graph



#Navpoints in the map = 100 – 5000



UT2004 World Abstraction

Underlaying classes – low level API

Classes of interest:

NavPoint, NavPointNeighbourLink, Item

ILocated, Location, DistanceUtils

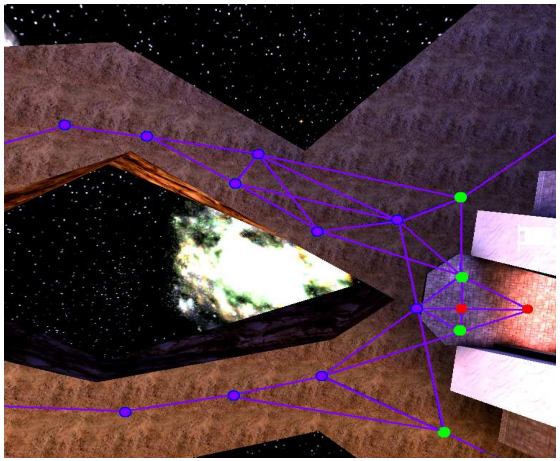
ItemType, ItemType.Category

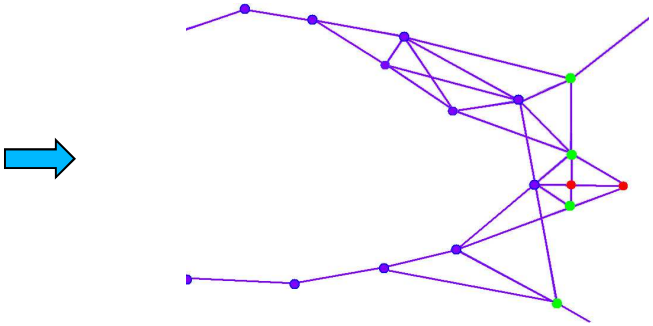
ItemDescriptor

Methods of interest:

```
this.items.getAllItems(ItemType)
this.descriptors.getDescriptor(ItemType)
this.world.getAll(NavPoint.class)
this.world.getAll(Item.class)
NavPoint.getOutgoingEdges()
NavPoint.getIncomingEdges()
```

!!!





UT2004 World Abstraction

NavPoint/NeighbourLink types

- **NavPoint types**
  - JumpPad
  - Lift
  - Teleport
  - Door
  - PlayerStart
  - SnipingSpot
  - InventorySpot
  - …
- **Link flags**
  - Walk
  - Jump
  - Lift
  - Door
  - DoubleJump
  - …

Today's menu

Navigating inside UT2004

1. Big Picture
2. PogamutUT2004 World Abstraction
3. **PogamutUT2004 Navigation**

Navigation

Step by step

Navigation steps:

1. Decide where to go
2. Plan the path (list of navpoints)
3. Follow the path
   - Handle jumps&lifts along the way!
     - Do you know right constants?
   - World is non-deterministic, be sure to check how the action is executing!
     => `IStuckDetector` implementations

Don’t worry it’s already wrapped up ☺

Navigation

Step by step

1. Decide where to go (Decision making!)
   - `items.getSpawnedItems(ItemType)`
   - `navPoints.getNavPoints()`
   - `DistanceUtils.getNearest(…)`
   - `MyCollections.getRandom(…)`
   - `fwMap.getNearest(…)`
2. + 3. Plan and follow the path
   - UT2004Navigation (this.navigation)

Navigation

UT2004Navigation

- Complete navigation wrapper
  - **`UT2004Navigation`**`(…, UT2004PathExecutor, FloydWarshallMap, …) (this.navigation)`
  - Handles both path planning & path following
  - Can be called repeatedly
  - Contains `this.pathExecutor`, `this.fwMap`
- Main methods
  - `navigation.navigate(…)`
  - `navigation.isNavigating()`
  - `navigation.stopNavigation()`
- Uses
  - `FloydWarshallMap (this.fwMap)`
  - `StuckDetectors`
  - `UT2004PathExecutor`

Navigation
FloydWarshallMap

- Pogamut path planner uses **Floyd Warshall** algorithm ($O(n^3)$!)
  - Used by `UT2004Navigation`
  - Access by `this.fwMap`
  - FW matrix is auto-initialized
- Methods of interest
  - `fwMap.getNearest…(…)`
    - Works the same as in `DistanceUtils`, except the distance is measured by the path length
    - Its ok to "spam" it (e.g. checking all items in each step), the nowadays computers can handle it

Navigation

Modifying the navigation graph

- NavigationGraphBuilder
  - Access by `this.navBuilder`
- Methods of interest
  - `navBuilder.removeEdgesBetween(…)`
- If you use `navBuilder` in **botInitalized** method, everything will be applied automatically
  - Otherwise, call `fwMap.refreshPathMatrix()`
    - `O(n³) !!`

Navigation
StuckDetectors

- Navigation Uses 3 stuck detectors
- **UT2004TimeStuckDetector(bot, 3000)**
  - if the bot does not move for 3 seconds consider it is stuck (check small velocity delta)
- **UT2004PositionStuckDetector()**
  - watch over the position history of the bot, if the bot does not move sufficiently enough, consider that it is stuck
  - DEFAULT_HISTORY_LENGTH, DEFAULT_MIN_DIAMETER, DEFAULT_MIN_Z
- **UT2004DistanceStuckDetector()**
  - counts how many times the bot was getting closer to the target and how many times it was getting farther (if it oscillates more than two times -> STUCK)

Navigation

Listening for navigation events

- With a FlagListener! Add one with method addStrongNavigationListener

```
this.navigation.addStrongNavigationListener(
                    new FlagListener<NavigationState>() {
                        @Override
                        public void flagChanged(NavigationState
   changedValue){
                            switch (changedValue) {
                                case STUCK:
                                break;
                                case STOPPED:
                                break;
                                case TARGET_REACHED:
                                break;
                                case PATH_COMPUTATION_FAILED:
                                break;
                                case NAVIGATING:
                                break;
                            }
                        }
        });
```

Navigation
Path following hell

- **UT2004PathExecutor**
- Custom Pogamut path following code
  - Heavily tweaked for UT2004 and game update frequency 4 Hz (250 ms per synchronous batch)
- The good
  - Works decently on non-complex maps
  - You don't have to do it yourself
- The bad
  - Has problems handling complex links
  - Spaghetti code

Navigation

Stuck detection details

- Inside `UT2004PathExecutorStuckState`
  - Who has detected the stuck
  - Which `NavPointNeighboutLink` bot failed to traverse
- Version: 3.5.1-SNAPSHOT and later

```
this.pathExecutor.getState().addStrongListener(
   new FlagListener<IPathExecutorState>() {
      @Override
      public void flagChanged(IPathExecutorState
                                                event) {
         switch (changedValue.getState()) {
            case STUCK:
               UT2004PathExecutorStuckState
                  stuckDetails =
                     (UT2004PathExecutorStuckState)
                     event;
               …
               break;
            …
         }
   }
});
```

Navigation – Legacy

Path planner & Path executor

1. Plan the path (list of navpoints)
   - `pathPlanner.computePath(ILocated from, to)`
     - Watch out for UT2004 quirks! Max 31 navpoints per path (+ starting position location == 32 path points).
   - `fwMap.computePath(NavPoint from, to)`
     - Plans path only between `NavPoints`
2. Follow the path
   - `pathExecutor.followPath(path)`
   - `pathExecutor.isExecuting()`
   - Watch out for its statefullness!

Navigation – Legacy

fwMap vs. pathPlanner

pathPlanner

- Path is planned at UT2004 => slower
- Graph is fixed
- May plan everywhere
- Has limit ~ 32 path points

fwMap

- Floyd-Warshall
  - O(n) path retrievel
- Graph may be altered
- Can't plan to all locations

`pathExecutor` works with both!

Assignment 5
Navigation Bot

- Let's create **NavigationBot**
  - Choose a NavPoint
  - Run to that NavPoint
  - Iterate
  - Handle bot stucking
- How to detect that the bot has stuck?
- What if the location is currently unreachable?
  - See `TabooSet` class
- Try it on DM-1on1-Albatross

```
ucc server DM-1on1-Albatross?game=GameBots2004.BotDeathMatch?...
```

Assignment 5
Cheatsheet

- Deciding where to go
  - `MyCollections.getRandom()`
  - `DistanceUtils…`
- Navigation module
  - `this.navigation.navigate(…)`
  - `this.navigation.isNavigating()`
- Stuck listening
  - ```
    this.navigation
          .addStrongNavigationListener(
    new FlagListener<NavigationState>() { …
    })
    ```
    ```

Send us finished assignment

Via e-mail:

- *Subject*
  - "Pogamut homework 2013 – Assignment X"
    - Replace `'X'` with the assignment number and the subject has to be without quotes of course
    - …or face -2 score penalization
- *To*
  - jakub.gemrot@gmail.com
    - Jakub Gemrot (Monday practice lessons)
  - michal.bida@gmail.com
    - Michal Bída (Thursday practice lessons)
- *Attachment*
  - Completely zip-up your project(s) folder except `'target'` directory and `IDE specific files` (or face -2 score penalization)
- *Body*
  - **Please send us information about how much time it took you to finish the assignment + any comments regarding your implementation struggle**
    - *Information won't be abused/made public*
    - *In fact it helps to make the practice lessons better*
  - Don't forget to mention your full name!

Questions?

I sense a soul in search of answers…

- We do not own the patent of perfection (yet…)
- In case of doubts about the assignment, tournament or hard problems, bugs don't hesitate to contact us!
  - Jakub Gemrot (Monday practice lessons)
    - jakub.gemrot@gmail.com
  - Michal Bída (Thursday practice lessons)
    - michal.bida@gmail.com

OPERAČNÍ PROGRAM PRAHA ADAPTABILITA

EVROPSKÁ UNIE

DĚKUJI ZA POZORNOST