King Fahd University of petroleum and minerals
Collage of Computer Science and Engineering

COE 484 - Robotics

Report on Head Control modules

GT 2005 is a great Simulator for AIBO RoboCup league. It is important to investigate whether parts of the code could be used to program a Kondo humanoid robot. Our goal is to program a Kondo smart enough that it could play soccer against other robots.

Zuhair Y. Khayyat
3/26/2008

Objective:

The objective of this report is to identify the relationship between "HeadControl" modules with both HeadControlMode and HeadMotionRequest representations. There are several modules that are used together to control the robot's head. Both header and class files can be found in "Modules/HeadControl/" folder. The existing files are as following:

1. HeadControl.h
2. HeadControlSelector.h
3. Xabsl2HeadControl.cpp
4. Xabsl2HeadControl.h
5. other head control files exists in "Modules/HeadControl/GT2005HeadControl/" folder

1. HeadControl.h:

It has two classes which are classHeadControlInterfaces and HeadControl. HeadControl class is used to calculate a new head motion which has to be set by the motion module based on the desired head mode and the current collection of percepts stored in the world state as well as the current sensor information. The constructor of HeadControl class access the robot configuration and its dimensions. HeadControl.h imports the following files:

Tools/Module/Module.h

Representations/Perception/SensorDataBuffer.h

Representations/Perception/BodyPosture.h

Representations/Perception/CameraMatrix.h

Representations/Cognition/LandmarksState.h

Representations/Cognition/RobotPose.h

Representations/Cognition/RobotState.h

Representations/Cognition/BallModel.h

Representations/Cognition/ObstaclesModel.h

Representations/Motion/OdometryData.h

Representations/Motion/HeadControlMode.h

Representations/Motion/HeadMotionRequest.h

Representations/Motion/PIDData.h

Representations/Motion/MotionInfo.h

Representations/RoboCup/GameControlData.h

Tools/Debugging/DebugDrawings.h

Tools/RobotConfiguration.h

Tools/RingBuffer.h"

Module.h, OdometryData.h, HeadControlMode.h HeadMotionRequest.h, PIDData.h, MotionInfo.h, GameControlData.h, RobotConfiguration.h and RingBuffer.h are important files for controlling the head. Eventhough the other files are also important, they will be investigated by other team members.

a) Module.h:

This header is the base class fore all modules which can be called from a MessageQueue to distribute messages to the rest of classes.

b) OdometryData.h:

OdometryData contains an approximation of overall movement the robot has done.

c) RobotConfiguration.h:

It is a class that represents the calibration and configuration of the robot. The file that holds the configuration of the robot is robot.cfg which is being read by RobotConfiguration class in RobotConfiguration.cpp.

d) HeadControlMode.h:

It stores head modes requested by BehaviorControl while HeadControlMode.cpp stores head modes requested by BehaviorControl. Such modes as following:

none	**directedScanForLandmarks**
searchForBall	**directedScanForObstacles**
searchAuto	**direct**
searchForLandmarks	**calibrate**
searchForLandmarksHeadLow	**calibrateHeadSpeed**
scanForObstacles	**watchOrigin**
lookAtMostInformativeLandmark	**openChallengePullBridge**
lookAtBluePinkLandmark	**openChallengeTest**

lookBetweenFeet	**openChallengeTest2**
lookLeft	**openChallengeGoToBridge**
lookRight	**openChallengeJoysickMode**
lookStraightAhead	**openChallengeReset**
catchBall	**searchForBallLeft**
lookBetweenFeetForCarriedBall	**searchForBallRight**
catchBallHigh	**lookAtInsertionPointBackLeft**
holdBall	**lookAtInsertionPointBackRight**
releaseCaughtBallWhenTurningLeft	**lookAtInsertionPointMiddleLeft**
releaseCaughtBallWhenTurningRight	**lookAtInsertionPointMiddleRight**
stayAsForced	**lookAtInsertionPointFrontLeft**
lookToStars	**lookAtInsertionPointFrontRight**
snapAtFinger	**lookAroundPropagatedBallPosition**
lookParallelToGround	**lookAroundCommunicatedBallPosition**
lookTowardOpponentGoal	**searchForLandmarksEvolution**
realSlowScan	

e) HeadMotionRequest.h:

Represents a motion request for the head that reads a HeadMotionRequest from a stream and writes a HeadMotionRequest to a stream. HeadMotionRequest.cpp Stores MotionRequests for the Head

f) PIDData.h:

It Contain servo gain values for all used joints (actuators).

g) MotionInfo.h:

It contains information about the motions which are executed by the Motion process

h) GameControlData.h:

Encapsulates the RoboCupGameControlData struct for the GT2005 project where GameControlData.cpp is the implementation. It has the following functions:

1- returns the name for the variable kickoff
2- returns the name of the current penalty
3- returns the penalty of the player
4- returns the number (01) of the data.teams -array the robot belongs to
5- returns a reference to the teamInfo of the team of the Player
6- returns the number (01) of the data.teams -array of the robots opponents
7- returns the robot-info for a player

8- returns a reference to the teamInfo of the team of the robots opponents

9- returns the Kickoff-Team as own/opponent

10- The timestamp when the last request was received

Moreover, the following states of the robot can be controlled by GameControlData.h:

1- initial

2- ready

3- set

4- playing

5- finished

6- unknown state

On the other hand, the following penalties states are also implemented:

1- notPenalized

2- illegalDefender

3- illegalDefense

4- obstruction

5- goaliePushing

6- playerPushing

7- ballHolding

8- requestForPickup

9- unknown penalty

i) RingBuffer.h:

It is some kind of a template data cyclic buffering.

2. HeadControlSelector.h:

This header is considered a selector for head control modules. It interface The paramters of the HeadControl module and handle them.

3. Xabsl2HeadControl.h:

This class is the base class for all HeadControl solutions that use a Xabsl2Engine. It interfaces The paramters of the HeadControl module to Xabsl2Engine. It can handle situation such as if the engine could not be created. It can also Allows to register the same Xabsl2HeadControl instance more than once at a ModuleHandler to be able to switch between different agents using the GT Module mechanism. It includes the XABSL engine from "Tools/Xabsl2/GT/GTXabsl2EngineExecutor.h". Xabsl2HeadControl.cpp is simply the implementation of the header.

4. "Modules/HeadControl/GT2005HeadControl/" files:

Those file are independent from all of the files exists directly in "Modules/HeadControl/" at the current level of investigation. No more investigation were done to files in "Modules/HeadControl/GT2005HeadControl/". Future investigation might occur depending on their role in controlling the robot's head.