Kinova Mico2 Windows Cartesian Controller

1.1.0

Generated by Doxygen 1.8.11

Contents

Chapter 1

Introduction

This project is a C++ implementation of a Cartesian controller that provides basic human-machine interface modules for real-time end-effector Cartesian position/velocity control of the Kinova Mico2 robot arm using the built-in velocity controllers in the Kinova API. The project is written in C++ and is intended to be run on Windows 8.1/10 with Visual Studio 2017 (v15). The code is distributed under the MIT license for maximum flexibility of use. By using this software, you are agreeing to the license. Please read the license prior to using this project.

This introduction is broken down into the following sections.

A note on units: All units in the library, unless specified, are in SI (international standard), i.e.: radians, meters, kilograms, etc...

Chapter 2

Software License

Chapter 3

Installation Notes

This project was built in Visual Studio 2017 (Community Edition) and the solution file is provided for reference. Additional dependencies need to be installed before building this project with the source and library files from the dependencies correctly linked to the project. Instructions for setting up the project and building it are presented below.

3.1 Install Dependencies

The following are the list of dependencies that are used in this project:

- Kinova Mico2 SDK [to control the Kinova Mico-2 robot arm]
- Kinect SDK v1.8 [to use the Microsoft Kinect sensor for skeleton tracking]
- CHAI3D (version 3.2.0) [to use the Novint Falcon Haptics controller]

3.1.1 Kinova Mico2 SDK

1. Download the SDK from the Kinova website and follow the instructions in the documentation to install the 32-bit version of the API.
2. Set the System Environment Variable KINOVASDK_DIR to the installation directory.

3.1.2 Kinect SDK v1.8

1. Download and install the Kinect v1.8 SDK from the Microsoft website and follow the instructions in the documentation to install the 32-but version of the API.
2. Check whether the system environment variable KINECTSDK10_DIR is set to the correct installation directory.

3.1.3 CHAI3D (version 3.2.0)

1. Download the multiplatform version of the CHAI3D SDK (currently tested with version 3.2.0) from the CHAI3D website
2. Install the dependencies (HDF5 and ZLIB) and build the appropriate Visual Studio Solution that matches the available edition.
3. Set the system environment variable CHAI3D_DIR to the installation directory

Chapter 4

Overview

Chapter 5

Getting Started

Chapter 6

References

Chapter 7

Class Index

7.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 8

File Index

8.1 File List

Here is a list of all files with brief descriptions:

Chapter 9

Class Documentation

9.1 CSocketInConnection Class Reference

`#include <socketInConnection.h>`

Public Member Functions

- CSocketInConnection (int theConnectionPort, unsigned short maxPacketSize=250, char headerID1=59, char headerID2=57)
- virtual ~CSocketInConnection ()
- bool connectToClient ()
- char * receiveData (int &dataSize)
- bool replyToReceivedData (char *data, int dataSize)
- std::string getConnectedMachineIP ()

Protected Member Functions

- bool _sendSimplePacket (char *packet, int packetLength, unsigned short packetsLeft)
- int _receiveSimplePacket (std::vector< char > &packet)
- unsigned int _getTimeInMs ()
- unsigned int _getTimeDiffInMs (unsigned int lastTime)

Protected Attributes

- SOCKET _socketServer
- SOCKET _socketClient
- struct sockaddr_in _socketLocal
- fd_set _socketTheSet
- int _socketConnectionPort
- bool _socketConnectWasOk
- std::string _socketConnectedMachineIP
- char _headerByte1
- char _headerByte2
- unsigned short _maxPacketSize

9.1.1 Constructor & Destructor Documentation

9.1.1.1 CSocketInConnection::CSocketInConnection (int *theConnectionPort,* unsigned short *maxPacketSize =* 250, char *headerID1 =* 59, char *headerID2 =* 57)

9.1.1.2 CSocketInConnection::~CSocketInConnection ( ) [virtual]

9.1.2 Member Function Documentation

9.1.2.1 unsigned int CSocketInConnection::_getTimeDiffInMs (unsigned int *lastTime*) [protected]

9.1.2.2 unsigned int CSocketInConnection::_getTimeInMs ( ) [protected]

9.1.2.3 int CSocketInConnection::_receiveSimplePacket (std::vector< char > & *packet*) [protected]

9.1.2.4 bool CSocketInConnection::_sendSimplePacket (char * *packet,* int *packetLength,* unsigned short *packetsLeft*) [protected]

9.1.2.5 bool CSocketInConnection::connectToClient ( )

9.1.2.6 std::string CSocketInConnection::getConnectedMachineIP ( )

9.1.2.7 char * CSocketInConnection::receiveData (int & *dataSize*)

9.1.2.8 bool CSocketInConnection::replyToReceivedData (char * *data,* int *dataSize*)

9.1.3 Member Data Documentation

9.1.3.1 char CSocketInConnection::_headerByte1 [protected]

9.1.3.2 char CSocketInConnection::_headerByte2 [protected]

9.1.3.3 unsigned short CSocketInConnection::_maxPacketSize [protected]

9.1.3.4 SOCKET CSocketInConnection::_socketClient [protected]

9.1.3.5 std::string CSocketInConnection::_socketConnectedMachineIP [protected]

9.1.3.6 int CSocketInConnection::_socketConnectionPort [protected]

9.1.3.7 bool CSocketInConnection::_socketConnectWasOk [protected]

9.1.3.8 struct sockaddr_in CSocketInConnection::_socketLocal [protected]

9.1.3.9 SOCKET CSocketInConnection::_socketServer [protected]

9.1.3.10 fd_set CSocketInConnection::_socketTheSet [protected]

The documentation for this class was generated from the following files:

- socketInConnection.h
- socketInConnection.cpp

9.2 CSocketOutConnection Class Reference

`#include <socketOutConnection.h>`

Public Member Functions

- CSocketOutConnection (const char *theConnectionAddress, int theConnectionPort, unsigned short maxPacketSize=250, char headerID1=59, char headerID2=57)
- virtual ~CSocketOutConnection ()
- int connectToServer ()
- bool sendData (char *data, int dataSize)
- char * receiveReplyData (int &dataSize)

Protected Member Functions

- bool _sendSimplePacket (char *packet, int packetLength, unsigned short packetsLeft)
- int _receiveSimplePacket (std::vector< char > &packet)
- int _getTimeInMs ()
- int _getTimeDiffInMs (int lastTime)

Protected Attributes

- std::string _socketConnectionAddress
- int _socketConnectionPort
- SOCKET _socketConn
- struct sockaddr_in _socketServer
- char _headerByte1
- char _headerByte2
- unsigned short _maxPacketSize

9.2.1 Constructor & Destructor Documentation

9.2.1.1 CSocketOutConnection::CSocketOutConnection (const char * *theConnectionAddress*, int *theConnectionPort*, unsigned short *maxPacketSize* = `250`, char *headerID1* = `59`, char *headerID2* = `57`)

9.2.1.2 virtual CSocketOutConnection::~CSocketOutConnection ( ) `[virtual]`

9.2.2 Member Function Documentation

9.2.2.1 int CSocketOutConnection::_getTimeDiffInMs (int *lastTime*) `[protected]`

9.2.2.2 int CSocketOutConnection::_getTimeInMs ( ) `[protected]`

9.2.2.3 int CSocketOutConnection::_receiveSimplePacket (std::vector< char > & *packet*) `[protected]`

9.2.2.4 bool CSocketOutConnection::_sendSimplePacket (char * *packet,* int *packetLength,* unsigned short *packetsLeft*) [protected]

9.2.2.5 int CSocketOutConnection::connectToServer ( )

9.2.2.6 char* CSocketOutConnection::receiveReplyData (int & *dataSize*)

9.2.2.7 bool CSocketOutConnection::sendData (char * *data,* int *dataSize*)

9.2.3 Member Data Documentation

9.2.3.1 char CSocketOutConnection::_headerByte1 [protected]

9.2.3.2 char CSocketOutConnection::_headerByte2 [protected]

9.2.3.3 unsigned short CSocketOutConnection::_maxPacketSize [protected]

9.2.3.4 SOCKET CSocketOutConnection::_socketConn [protected]

9.2.3.5 std::string CSocketOutConnection::_socketConnectionAddress [protected]

9.2.3.6 int CSocketOutConnection::_socketConnectionPort [protected]

9.2.3.7 struct sockaddr_in CSocketOutConnection::_socketServer [protected]

The documentation for this class was generated from the following file:

- socketOutConnection.h

9.3 CXBOXController Class Reference

```
#include <CartesianControl.h>
```

Public Member Functions

- CXBOXController (int playerNumber)
- XINPUT_STATE GetState ()
- bool IsConnected ()
- void Vibrate (int leftVal=0, int rightVal=0)

9.3.1 Constructor & Destructor Documentation

9.3.1.1 CXBOXController::CXBOXController (int *playerNumber*)

9.3.2 Member Function Documentation

9.3.2.1 XINPUT_STATE CXBOXController::GetState ( )

9.3.2.2 bool CXBOXController::IsConnected ( )

9.3.2.3 void CXBOXController::Vibrate (int *leftVal* = 0, int *rightVal* = 0)

The documentation for this class was generated from the following file:

- CartesianControl.h

9.4 Experiment Class Reference

```
#include <CartesianControl.h>
```

Public Member Functions

- void HoloLensCartesianTeleop (char ∗argv[ ], KinovaAPIFunctions kinova)
- double interp_lut (double t, double ∗y_lut, int L_lut, double Tp_lut, double fs_lut)
- void MovetoStartPos (KinovaAPIFunctions kinova)
- void MoveEndEffectorPos (KinovaAPIFunctions kinova, float xe, float ze)
- bool load_LUT1D (char ∗filename, char delim)

Public Attributes

- double T = 10.0
- double T_calib = 10.0
- double T_TF = 10.0
- double Ts = 1.0 / 125.0
- const double fs_qd_lut = 3000.0
- const double fs_uc_lut = 3000.0
- double scale_LUT = 1
- double ∗ LUT
- int len_LUT
- const float Kp = 2.0f

9.4.1 Member Function Documentation

9.4.1.1 void Experiment::HoloLensCartesianTeleop (char * *argv[ ],* KinovaAPIFunctions *kinova*)

9.4.1.2 double Experiment::interp_lut (double *t,* double * *y_lut,* int *L_lut,* double *Tp_lut,* double *fs_lut*)

This function linearly interpolates the values in the Lookup table for a given input time instant.

Inputs:

1. (double) t : current time instant (s)
2. (double*) ylut : Lookup Table
3. (int) L_lut : Length of the Lookup table
4. (double) Tp_lut : Time period of Lookup table (s)
5. (double) fs_lut : Sampling frequency of Lookupt table (Hz)

Output:

1. (double) : output of interpolation

9.4.1.3 bool Experiment::load_LUT1D (char * *filename,* char *delim*)

Load Lookup table from CSV file

Inputs:

1. (char*) filename: Filename of CSV file
2. (char) delim : Delimiter for CSV file

Output:

1. (bool) : Status of LUT load operation

9.4.1.4 void Experiment::MoveEndEffectorPos (KinovaAPIFunctions *kinova,* float *xe,* float *ze*)

9.4.1.5 void Experiment::MovetoStartPos (KinovaAPIFunctions *kinova*)

9.4.2 Member Data Documentation

9.4.2.1 const double Experiment::fs_qd_lut = 3000.0

9.4.2.2 const double Experiment::fs_uc_lut = 3000.0

9.4.2.3 const float Experiment::Kp = 2.0f

9.4.2.4 int Experiment::len_LUT

9.4.2.5 double* Experiment::LUT

9.4.2.6 double Experiment::scale_LUT = 1

9.4.2.7 double Experiment::T = 10.0

9.4.2.8 double Experiment::T_calib = 10.0

9.4.2.9 double Experiment::T_TF = 10.0

9.4.2.10 double Experiment::Ts = 1.0 / 125.0

The documentation for this class was generated from the following files:

- CartesianControl.h
- angularCommandControl.cpp
- expt_HoloLensCartesianTeleop.cpp
- lutLinearInterp.cpp

9.5 FIRFilter Class Reference

`#include <CartesianControl.h>`

Public Member Functions

- bool firFloatInit ()

  Function to initialize the FIR Filter.
- double * firFloat (double *input, int length)

  Function to compute the output of the FIR Filter.

Public Attributes

- const char * filename = "./lowpass.mat"

  Filename of FIR Filter weights.
- int length = 1
- double * output = new double[length]
- double insamp [BUFFER_LEN]

  buffer array to hold input samples
- double * coeffs

  FIR Filter weights.
- double cutoff_freq_hz = 0.2

  Cutoff frequency of the FIR Filter in Hz.

9.5.1 Member Function Documentation

9.5.1.1 double * FIRFilter::firFloat (double * *input*, int *length*)

Function to compute the output of the FIR Filter.

FIR Filter Computation

This function computes the output of the FIR Filter for the given content of the input buffer in three steps:

1) Put new input at the high end of the buffer
2) Apply the filter to each input sample
3) Shift input samples back in time for next time instant

9.5.1.2 bool FIRFilter::firFloatInit ( )

Function to initialize the FIR Filter.

FIR Filter Initialization

This function initializes the FIR filter by searching for a CSV file formatted as

num_weights
w1
w2
.
.
.
wn
which is read into the coeffs variable.

9.5.2 Member Data Documentation

9.5.2.1 double* FIRFilter::coeffs

FIR Filter weights.

9.5.2.2 double FIRFilter::cutoff_freq_hz = 0.2

Cutoff frequency of the FIR Filter in Hz.

9.5.2.3 const char* FIRFilter::filename = "./lowpass.mat"

Filename of FIR Filter weights.

9.5.2.4 double FIRFilter::insamp[BUFFER_LEN]

buffer array to hold input samples

9.5.2.5 int FIRFilter::length = 1

9.5.2.6 double* FIRFilter::output = new double[length]

The documentation for this class was generated from the following files:

- CartesianControl.h
- firFilter.cpp

9.6 kinectSkelTrack::KinectInfo Struct Reference

`#include <CartesianControl.h>`

Public Attributes

- bool startSignal = false
- bool userFound = false
- double handPosition = 0

9.6.1 Member Data Documentation

9.6.1.1 double kinectSkelTrack::KinectInfo::handPosition = 0

9.6.1.2 bool kinectSkelTrack::KinectInfo::startSignal = false

9.6.1.3 bool kinectSkelTrack::KinectInfo::userFound = false

The documentation for this struct was generated from the following file:

- CartesianControl.h

9.7 kinectSkelTrack Class Reference

`#include <CartesianControl.h>`

Classes

- struct KinectInfo

Public Member Functions

- bool initKinect ()
- void getSkeletalData ()
- KinectInfo getKinectData ()

Public Attributes

- int width = 640
- int height = 480
- HANDLE depthStream
- INuiSensor * sensor
- Vector4 skeletonPosition [NUI_SKELETON_POSITION_COUNT]

9.7.1 Member Function Documentation

9.7.1.1 kinectSkelTrack::KinectInfo kinectSkelTrack::getKinectData ( )

Auxiliary function if multithreading the Kinect sensor

9.7.1.2 void kinectSkelTrack::getSkeletalData ( )

Get the skeletal data from the current frame

9.7.1.3 bool kinectSkelTrack::initKinect ( )

Initialize the Kinect sensor for skeleton tracking of the user closest to the sensor.

9.7.2 Member Data Documentation

9.7.2.1 HANDLE kinectSkelTrack::depthStream

9.7.2.2 int kinectSkelTrack::height = 480

9.7.2.3 INuiSensor* kinectSkelTrack::sensor

9.7.2.4 Vector4 kinectSkelTrack::skeletonPosition[NUI_SKELETON_POSITION_COUNT]

9.7.2.5 int kinectSkelTrack::width = 640

The documentation for this class was generated from the following files:

- CartesianControl.h
- HMIfunctions.cpp

9.8 KinovaAPIFunctions Class Reference

`#include <CartesianControl.h>`

Public Attributes

- int(∗ MyInitAPI)()
- int(∗ MyCloseAPI)()
- int(∗ MyStartControlAPI)()
- int(∗ MyStartForceControl)()
- int(∗ MyStopForceControl)()
- int(∗ MyGetClientConfigurations)(ClientConfigurations &config)
- int(∗ MySetClientConfigurations)(ClientConfigurations config)
- int(∗ MySendJoystickCommand)(JoystickCommand joystickCommand)
- int(∗ MyGetGlobalTrajectoryInfo)(TrajectoryFIFO &Response)
- int(∗ MyRunGravityZEstimationSequence)(ROBOT_TYPE type, double OptimalzParam[OPTIMAL_Z_PA↩ RAM_SIZE])
- int(∗ MySendBasicTrajectory)(TrajectoryPoint command)
- int(∗ MyGetActualTrajectoryInfo)(TrajectoryPoint &)
- int(∗ MyGetDevices)(KinovaDevice devices[MAX_KINOVA_DEVICE], int &result)
- int(∗ MySetActiveDevice)(KinovaDevice device)
- int(∗ MySetGravityOptimalZParam)(double optimalZParams[OPTIMAL_Z_PARAM_SIZE])
- int(∗ MySetGravityType)(GRAVITY_TYPE type)
- int(∗ MyMoveHome)()
- int(∗ MyInitFingers)()
- int(∗ MyGetCartesianCommand)(CartesianPosition &)
- int(∗ MyGetCartesianPosition)(CartesianPosition &)
- int(∗ MyGetAngularPosition)(AngularPosition &)
- int(∗ MySetActuatorPID)(unsigned int adress, float P, float I, float D)
- int(∗ MyGetGripperStatus)(Gripper &)

9.8.1 Member Data Documentation

9.8.1.1 int(∗ KinovaAPIFunctions::MyCloseAPI) ()

9.8.1.2 int(∗ KinovaAPIFunctions::MyGetActualTrajectoryInfo) (TrajectoryPoint &)

9.8.1.3 int(∗ KinovaAPIFunctions::MyGetAngularPosition) (AngularPosition &)

9.8.1.4 int(∗ KinovaAPIFunctions::MyGetCartesianCommand) (CartesianPosition &)

9.8.1.5 int(∗ KinovaAPIFunctions::MyGetCartesianPosition) (CartesianPosition &)

9.8.1.6 int(∗ KinovaAPIFunctions::MyGetClientConfigurations) (ClientConfigurations &config)

9.8.1.7 int(∗ KinovaAPIFunctions::MyGetDevices) (KinovaDevice devices[MAX_KINOVA_DEVICE], int &result)

9.8.1.8 int(∗ KinovaAPIFunctions::MyGetGlobalTrajectoryInfo) (TrajectoryFIFO &Response)

9.8.1.9 int(* KinovaAPIFunctions::MyGetGripperStatus) (Gripper &)

9.8.1.10 int(* KinovaAPIFunctions::MyInitAPI) ()

9.8.1.11 int(* KinovaAPIFunctions::MyInitFingers) ()

9.8.1.12 int(* KinovaAPIFunctions::MyMoveHome) ()

9.8.1.13 int(* KinovaAPIFunctions::MyRunGravityZEstimationSequence) (ROBOT_TYPE type, double OptimalzParam[OPTIMAL_Z_PARAM_SIZE])

9.8.1.14 int(* KinovaAPIFunctions::MySendBasicTrajectory) (TrajectoryPoint command)

9.8.1.15 int(* KinovaAPIFunctions::MySendJoystickCommand) (JoystickCommand joystickCommand)

9.8.1.16 int(* KinovaAPIFunctions::MySetActiveDevice) (KinovaDevice device)

9.8.1.17 int(* KinovaAPIFunctions::MySetActuatorPID) (unsigned int adress, float P, float I, float D)

9.8.1.18 int(* KinovaAPIFunctions::MySetClientConfigurations) (ClientConfigurations config)

9.8.1.19 int(* KinovaAPIFunctions::MySetGravityOptimalZParam) (double optimalZParams[OPTIMAL_Z_PARAM_SIZE])

9.8.1.20 int(* KinovaAPIFunctions::MySetGravityType) (GRAVITY_TYPE type)

9.8.1.21 int(* KinovaAPIFunctions::MyStartControlAPI) ()

9.8.1.22 int(* KinovaAPIFunctions::MyStartForceControl) ()

9.8.1.23 int(* KinovaAPIFunctions::MyStopForceControl) ()

The documentation for this class was generated from the following file:

- CartesianControl.h

9.9 NovintFalconHapticsDevice Class Reference

`#include <CartesianControl.h>`

Public Member Functions

- bool InitializeHapticsDevice ()
- void UpdateHaptics (void)
- void ResetIC (void)

Public Attributes

- chai3d::cGenericHapticDevicePtr hapticDevice
- chai3d::cVector3d hapticDevicePosition
- chai3d::cThread * hapticsThread
- chai3d::cHapticDeviceHandler * handler
- double maxforce = 1
- chai3d::cVector3d position
- chai3d::cMatrix3d rotation
- chai3d::cVector3d linearVelocity
- chai3d::cVector3d angularVelocity
- chai3d::cVector3d desiredPosition
- chai3d::cMatrix3d desiredRotation
- bool isRunning = false
- bool button0_state = false
- bool button2_state = false
- bool useDamping = false
- bool useForceField = false

9.9.1 Member Function Documentation

9.9.1.1 bool NovintFalconHapticsDevice::InitializeHapticsDevice ( )

Initialize the Novint Falcon Haptics device

9.9.1.2 void NovintFalconHapticsDevice::ResetIC (void)

Reset the Haptics Device and apply a feedback force towards the center of the joystick

9.9.1.3 void NovintFalconHapticsDevice::UpdateHaptics (void)

Update the current position of the Haptics joystick

9.9.2 Member Data Documentation

9.9.2.1 chai3d::cVector3d NovintFalconHapticsDevice::angularVelocity

9.9.2.2 bool NovintFalconHapticsDevice::button0_state = false

9.9.2.3 bool NovintFalconHapticsDevice::button2_state = false

9.9.2.4 chai3d::cVector3d NovintFalconHapticsDevice::desiredPosition

9.9.2.5 chai3d::cMatrix3d NovintFalconHapticsDevice::desiredRotation

9.9.2.6 chai3d::cHapticDeviceHandler* NovintFalconHapticsDevice::handler

9.9.2.7 chai3d::cGenericHapticDevicePtr NovintFalconHapticsDevice::hapticDevice

9.9.2.8 chai3d::cVector3d NovintFalconHapticsDevice::hapticDevicePosition

9.9.2.9 chai3d::cThread* NovintFalconHapticsDevice::hapticsThread

9.9.2.10 bool NovintFalconHapticsDevice::isRunning = false

9.9.2.11 chai3d::cVector3d NovintFalconHapticsDevice::linearVelocity

9.9.2.12 double NovintFalconHapticsDevice::maxforce = 1

9.9.2.13 chai3d::cVector3d NovintFalconHapticsDevice::position

9.9.2.14 chai3d::cMatrix3d NovintFalconHapticsDevice::rotation

9.9.2.15 bool NovintFalconHapticsDevice::useDamping = false

9.9.2.16 bool NovintFalconHapticsDevice::useForceField = false

The documentation for this class was generated from the following files:

- CartesianControl.h
- HMIfunctions.cpp

9.10 Timer Class Reference

`#include <CartesianControl.h>`

Public Member Functions

- void start ()
- double elapsedTime ()
- bool isTimeout (unsigned long seconds)

9.10.1 Member Function Documentation

9.10.1.1 double Timer::elapsedTime ( ) [inline]

9.10.1.2 bool Timer::isTimeout (unsigned long *seconds*) [inline]

9.10.1.3 void Timer::start ( ) [inline]

The documentation for this class was generated from the following file:

- CartesianControl.h

Chapter 10

File Documentation

10.1 angularCommandControl.cpp File Reference

```
#include "stdafx.h"
#include "CartesianControl.h"
```

Include dependency graph for angularCommandControl.cpp:

Macros

- #define _WINSOCK_DEPRECATED_NO_WARNINGS
- #define WIN32_LEAN_AND_MEAN

10.1.1 Macro Definition Documentation

10.1.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS

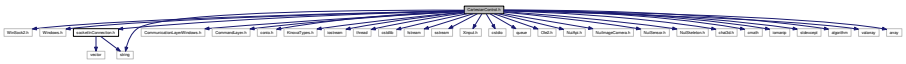
10.1.1.2 #define WIN32_LEAN_AND_MEAN

10.2 CartesianControl.h File Reference

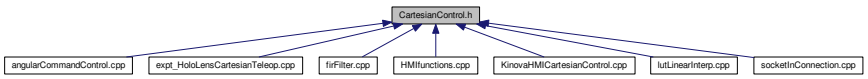
```
#include <WinSock2.h>
```

```
#include <Windows.h>
#include "socketInConnection.h"
#include "CommunicationLayerWindows.h"
#include "CommandLayer.h"
#include <conio.h>
#include "KinovaTypes.h"
#include <iostream>
#include <thread>
#include <cstdlib>
#include <fstream>
#include <string>
#include <sstream>
#include <Xinput.h>
#include <cstdio>
#include <queue>
#include <Ole2.h>
#include "NuiApi.h"
#include "NuiImageCamera.h"
#include "NuiSensor.h"
#include "NuiSkeleton.h"
#include "chai3d.h"
#include <cmath>
#include <iomanip>
#include <stdexcept>
#include <algorithm>
#include <valarray>
#include <array>
```

Include dependency graph for CartesianControl.h:



This graph shows which files directly or indirectly include this file:



Classes

- class CXBOXController
- class Timer
- class KinovaAPIFunctions
- class kinectSkelTrack
- struct kinectSkelTrack::KinectInfo
- class FIRFilter
- class Experiment
- class NovintFalconHapticsDevice

Macros

- #define _WINSOCK_DEPRECATED_NO_WARNINGS
- #define WIN32_LEAN_AND_MEAN
- #define WIN32_LEAN_AND_MEAN
- #define USE_KINOVA true
- #define USE_KINECT false
- #define USE_TCP true
- #define USE_HAPTICS true
- #define GNUPLOT false
- #define _XBOX_CONTROLLER_H_
- #define WIN32_LEAN_AND_MEAN
- #define BUFFER_LEN 96

  Buffer length to hold input samples of FIR Filter.

10.2.1 Macro Definition Documentation

10.2.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.2.1.2 #define _XBOX_CONTROLLER_H_

10.2.1.3 #define BUFFER_LEN 96

Buffer length to hold input samples of FIR Filter.

10.2.1.4 #define GNUPLOT false

10.2.1.5 #define USE_HAPTICS true

10.2.1.6 #define USE_KINECT false

10.2.1.7 #define USE_KINOVA true

10.2.1.8 #define USE_TCP true

10.2.1.9 #define WIN32_LEAN_AND_MEAN

10.2.1.10 #define WIN32_LEAN_AND_MEAN

10.2.1.11 #define WIN32_LEAN_AND_MEAN

10.3 expt_HoloLensCartesianTeleop.cpp File Reference

```
#include "stdafx.h"
#include "CartesianControl.h"
#include <vector>
#include <queue>
#include <future>
```

Include dependency graph for expt_HoloLensCartesianTeleop.cpp:

Macros

- #define TRAIN true
- #define TEST_1 false
- #define TEST_2 false
- #define INVERSE_CONTROLLER false
- #define LOAD_UHX_FROM_FILE false
- #define LOAD_XM_FROM_FILE false

Variables

- NovintFalconHapticsDevice * novintFalcon = new NovintFalconHapticsDevice

10.3.1 Macro Definition Documentation

10.3.1.1 #define INVERSE_CONTROLLER false

10.3.1.2 #define LOAD_UHX_FROM_FILE false

10.3.1.3 #define LOAD_XM_FROM_FILE false

10.3.1.4 #define TEST_1 false

10.3.1.5 #define TEST_2 false

10.3.1.6 #define TRAIN true

10.3.2 Variable Documentation

10.3.2.1 NovintFalconHapticsDevice* novintFalcon = new NovintFalconHapticsDevice

10.4 firFilter.cpp File Reference

```
#include "stdafx.h"
#include "CartesianControl.h"
```

Include dependency graph for firFilter.cpp:

Macros

- #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.4.1 Macro Definition Documentation

10.4.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.5 HMIfunctions.cpp File Reference

```
#include "stdafx.h"
#include "CartesianControl.h"
#include <future>
```

Include dependency graph for HMIfunctions.cpp:

Macros

- #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.5.1 Macro Definition Documentation

10.5.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.6 KinovaHMICartesianControl.cpp File Reference

```
#include "CartesianControl.h"
```

Include dependency graph for KinovaHMICartesianControl.cpp:

Functions

- int main (int argc, char *argv[])

Variables

- HINSTANCE commandLayer_handle

10.6.1 Function Documentation

10.6.1.1 int main (int *argc*, char * *argv[]*)

10.6.2 Variable Documentation

10.6.2.1 HINSTANCE commandLayer_handle

10.7 lutLinearInterp.cpp File Reference

```
#include "stdafx.h"
#include "CartesianControl.h"
```

Include dependency graph for lutLinearInterp.cpp:

Macros

- #define _WINSOCK_DEPRECATED_NO_WARNINGS
- #define WIN32_LEAN_AND_MEAN

10.7.1 Macro Definition Documentation

10.7.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.7.1.2 #define WIN32_LEAN_AND_MEAN

10.8 socketInConnection.cpp File Reference

```
#include "stdafx.h"
#include "socketInConnection.h"
#include "CartesianControl.h"
```

Include dependency graph for socketInConnection.cpp:

Macros

- #define _WINSOCK_DEPRECATED_NO_WARNINGS
- #define HEADER_LENGTH 6

10.8.1 Macro Definition Documentation

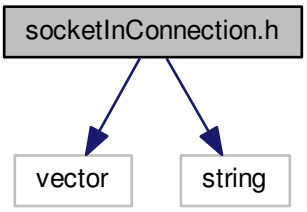
10.8.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS

10.8.1.2 #define HEADER_LENGTH 6

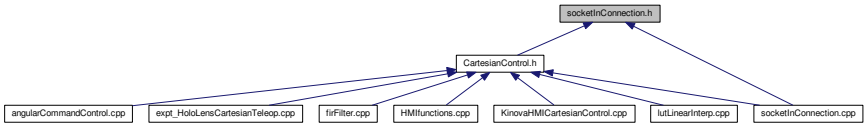
10.9 socketInConnection.h File Reference

```
#include <vector>
#include <string>
```

Include dependency graph for socketInConnection.h:



This graph shows which files directly or indirectly include this file:



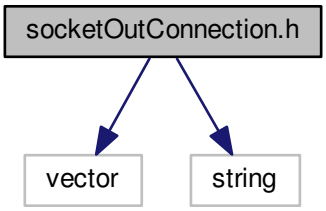
Classes

- class CSocketInConnection

10.10 socketOutConnection.h File Reference

```
#include <vector>
#include <string>
```

Include dependency graph for socketOutConnection.h:



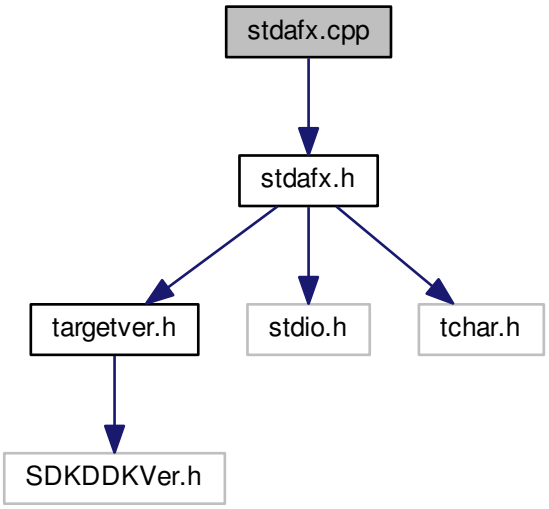
Classes

- class CSocketOutConnection

10.11 stdafx.cpp File Reference

```
#include "stdafx.h"
```

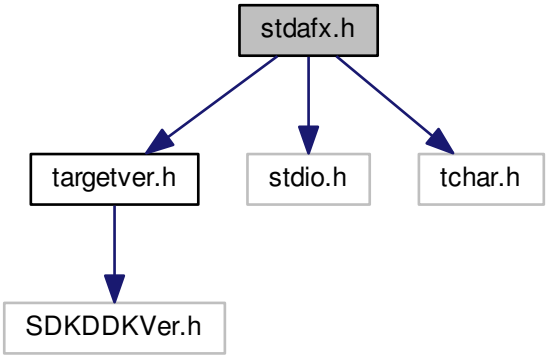
Include dependency graph for stdafx.cpp:



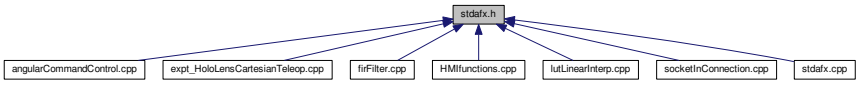
10.12 stdafx.h File Reference

```
#include "targetver.h"
#include <stdio.h>
#include <tchar.h>
```

Include dependency graph for stdafx.h:



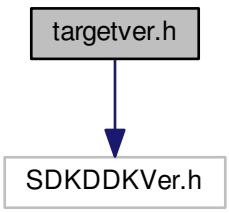
This graph shows which files directly or indirectly include this file:



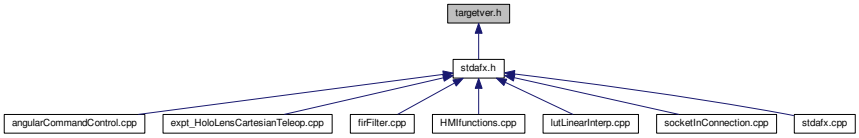
10.13 targetver.h File Reference

```
#include <SDKDDKVer.h>
```

Include dependency graph for targetver.h:



This graph shows which files directly or indirectly include this file:



Index