

# Kinova Mico2 Windows Cartesian Controller

## 1.1.0

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Software License</b>	<b>3</b>
<b>3</b>	<b>Installation Notes</b>	<b>5</b>
3.1	Install Dependencies . . . . .	5
3.1.1	Kinova Mico2 SDK . . . . .	5
3.1.2	Kinect SDK v1.8 . . . . .	5
3.1.3	CHAI3D (version 3.2.0) . . . . .	5
<b>4</b>	<b>Overview</b>	<b>7</b>
<b>5</b>	<b>Getting Started</b>	<b>9</b>
<b>6</b>	<b>References</b>	<b>11</b>
<b>7</b>	<b>Class Index</b>	<b>13</b>
7.1	Class List . . . . .	13
<b>8</b>	<b>File Index</b>	<b>15</b>
8.1	File List . . . . .	15

<b>9 Class Documentation</b>	<b>17</b>
9.1 CSocketInConnection Class Reference	17
9.1.1 Constructor & Destructor Documentation	18
9.1.1.1 CSocketInConnection(int theConnectionPort, unsigned short maxPacketSize=250, char headerID1=59, char headerID2=57)	18
9.1.1.2 ~CSocketInConnection()	18
9.1.2 Member Function Documentation	18
9.1.2.1 _getTimeDiffInMs(unsigned int lastTime)	18
9.1.2.2 _getTimeInMs()	18
9.1.2.3 _receiveSimplePacket(std::vector< char > &packet)	18
9.1.2.4 _sendSimplePacket(char *packet, int packetLength, unsigned short packetsLeft)	18
9.1.2.5 connectToClient()	18
9.1.2.6 getConnectedMachineIP()	18
9.1.2.7 receiveData(int &dataSize)	18
9.1.2.8 replyToReceivedData(char *data, int dataSize)	18
9.1.3 Member Data Documentation	18
9.1.3.1 _headerByte1	18
9.1.3.2 _headerByte2	18
9.1.3.3 _maxPacketSize	18
9.1.3.4 _socketClient	18
9.1.3.5 _socketConnectedMachineIP	18
9.1.3.6 _socketConnectionPort	18
9.1.3.7 _socketConnectWasOk	18
9.1.3.8 _socketLocal	18
9.1.3.9 _socketServer	18
9.1.3.10 _socketTheSet	18
9.2 CSocketOutConnection Class Reference	19
9.2.1 Constructor & Destructor Documentation	19
9.2.1.1 CSocketOutConnection(const char *theConnectionAddress, int theConnectionPort, unsigned short maxPacketSize=250, char headerID1=59, char headerID2=57)	19
9.2.1.2 ~CSocketOutConnection()	19

9.2.2	Member Function Documentation	19
9.2.2.1	<code>_getTimeDiffInMs(int lastTime)</code>	19
9.2.2.2	<code>_getTimeInMs()</code>	19
9.2.2.3	<code>_receiveSimplePacket(std::vector&lt; char &gt; &amp;packet)</code>	19
9.2.2.4	<code>_sendSimplePacket(char *packet, int packetLength, unsigned short packetsLeft)</code>	20
9.2.2.5	<code>connectToServer()</code>	20
9.2.2.6	<code>receiveReplyData(int &amp;dataSize)</code>	20
9.2.2.7	<code>sendData(char *data, int dataSize)</code>	20
9.2.3	Member Data Documentation	20
9.2.3.1	<code>_headerByte1</code>	20
9.2.3.2	<code>_headerByte2</code>	20
9.2.3.3	<code>_maxPacketSize</code>	20
9.2.3.4	<code>_socketConn</code>	20
9.2.3.5	<code>_socketConnectionAddress</code>	20
9.2.3.6	<code>_socketConnectionPort</code>	20
9.2.3.7	<code>_socketServer</code>	20
9.3	CXBOXController Class Reference	20
9.3.1	Constructor & Destructor Documentation	21
9.3.1.1	<code>CXBOXController(int playerNumber)</code>	21
9.3.2	Member Function Documentation	21
9.3.2.1	<code>GetState()</code>	21
9.3.2.2	<code>IsConnected()</code>	21
9.3.2.3	<code>Vibrate(int leftVal=0, int rightVal=0)</code>	21
9.4	Experiment Class Reference	21
9.4.1	Member Function Documentation	22
9.4.1.1	<code>HoloLensCartesianTeleop(char *argv[], KinovaAPIFunctions kinova)</code>	22
9.4.1.2	<code>interp_lut(double t, double *y_lut, int L_lut, double Tp_lut, double fs_lut)</code>	22
9.4.1.3	<code>load_LUT1D(char *filename, char delim)</code>	22
9.4.1.4	<code>MoveEndEffectorPos(KinovaAPIFunctions kinova, float xe, float ze)</code>	23
9.4.1.5	<code>MovetoStartPos(KinovaAPIFunctions kinova)</code>	23

9.4.2	Member Data Documentation . . . . .	23
9.4.2.1	fs_qd_lut . . . . .	23
9.4.2.2	fs_uc_lut . . . . .	23
9.4.2.3	Kp . . . . .	23
9.4.2.4	len_LUT . . . . .	23
9.4.2.5	LUT . . . . .	23
9.4.2.6	scale_LUT . . . . .	23
9.4.2.7	T . . . . .	23
9.4.2.8	T_calib . . . . .	23
9.4.2.9	T_TF . . . . .	23
9.4.2.10	Ts . . . . .	23
9.5	FIRFilter Class Reference . . . . .	23
9.5.1	Member Function Documentation . . . . .	24
9.5.1.1	firFloat(double *input, int length) . . . . .	24
9.5.1.2	firFloatInit() . . . . .	24
9.5.2	Member Data Documentation . . . . .	24
9.5.2.1	coeffs . . . . .	24
9.5.2.2	cutoff_freq_hz . . . . .	25
9.5.2.3	filename . . . . .	25
9.5.2.4	insamp . . . . .	25
9.5.2.5	length . . . . .	25
9.5.2.6	output . . . . .	25
9.6	kinectSkelTrack::KinectInfo Struct Reference . . . . .	25
9.6.1	Member Data Documentation . . . . .	25
9.6.1.1	handPosition . . . . .	25
9.6.1.2	startSignal . . . . .	25
9.6.1.3	userFound . . . . .	25
9.7	kinectSkelTrack Class Reference . . . . .	26
9.7.1	Member Function Documentation . . . . .	26
9.7.1.1	getKinectData() . . . . .	26

9.7.1.2	<a href="#">getSkeletalData()</a>	26
9.7.1.3	<a href="#">initKinect()</a>	26
9.7.2	<a href="#">Member Data Documentation</a>	26
9.7.2.1	<a href="#">depthStream</a>	26
9.7.2.2	<a href="#">height</a>	26
9.7.2.3	<a href="#">sensor</a>	26
9.7.2.4	<a href="#">skeletonPosition</a>	26
9.7.2.5	<a href="#">width</a>	26
9.8	<a href="#">KinovaAPIFunctions Class Reference</a>	27
9.8.1	<a href="#">Member Data Documentation</a>	27
9.8.1.1	<a href="#">MyCloseAPI</a>	27
9.8.1.2	<a href="#">MyGetActualTrajectoryInfo</a>	27
9.8.1.3	<a href="#">MyGetAngularPosition</a>	27
9.8.1.4	<a href="#">MyGetCartesianCommand</a>	27
9.8.1.5	<a href="#">MyGetCartesianPosition</a>	27
9.8.1.6	<a href="#">MyGetClientConfigurations</a>	27
9.8.1.7	<a href="#">MyGetDevices</a>	27
9.8.1.8	<a href="#">MyGetGlobalTrajectoryInfo</a>	27
9.8.1.9	<a href="#">MyGetGripperStatus</a>	28
9.8.1.10	<a href="#">MyInitAPI</a>	28
9.8.1.11	<a href="#">MyInitFingers</a>	28
9.8.1.12	<a href="#">MyMoveHome</a>	28
9.8.1.13	<a href="#">MyRunGravityZEstimationSequence</a>	28
9.8.1.14	<a href="#">MySendBasicTrajectory</a>	28
9.8.1.15	<a href="#">MySendJoystickCommand</a>	28
9.8.1.16	<a href="#">MySetActiveDevice</a>	28
9.8.1.17	<a href="#">MySetActuatorPID</a>	28
9.8.1.18	<a href="#">MySetClientConfigurations</a>	28
9.8.1.19	<a href="#">MySetGravityOptimalZParam</a>	28
9.8.1.20	<a href="#">MySetGravityType</a>	28

9.8.1.21	MyStartControlAPI . . . . .	28
9.8.1.22	MyStartForceControl . . . . .	28
9.8.1.23	MyStopForceControl . . . . .	28
9.9	NovintFalconHapticsDevice Class Reference . . . . .	28
9.9.1	Member Function Documentation . . . . .	29
9.9.1.1	InitializeHapticsDevice() . . . . .	29
9.9.1.2	ResetIC(void) . . . . .	29
9.9.1.3	UpdateHaptics(void) . . . . .	29
9.9.2	Member Data Documentation . . . . .	29
9.9.2.1	angularVelocity . . . . .	29
9.9.2.2	button0_state . . . . .	29
9.9.2.3	button2_state . . . . .	29
9.9.2.4	desiredPosition . . . . .	29
9.9.2.5	desiredRotation . . . . .	29
9.9.2.6	handler . . . . .	29
9.9.2.7	hapticDevice . . . . .	30
9.9.2.8	hapticDevicePosition . . . . .	30
9.9.2.9	hapticsThread . . . . .	30
9.9.2.10	isRunning . . . . .	30
9.9.2.11	linearVelocity . . . . .	30
9.9.2.12	maxforce . . . . .	30
9.9.2.13	position . . . . .	30
9.9.2.14	rotation . . . . .	30
9.9.2.15	useDamping . . . . .	30
9.9.2.16	useForceField . . . . .	30
9.10	Timer Class Reference . . . . .	30
9.10.1	Member Function Documentation . . . . .	30
9.10.1.1	elapsedTime() . . . . .	30
9.10.1.2	isTimeout(unsigned long seconds) . . . . .	30
9.10.1.3	start() . . . . .	30



<b>10 File Documentation</b>	<b>31</b>
10.1 angularCommandControl.cpp File Reference	31
10.1.1 Macro Definition Documentation	31
10.1.1.1 _WINSOCK_DEPRECATED_NO_WARNINGS	31
10.1.1.2 WIN32_LEAN_AND_MEAN	31
10.2 CartesianControl.h File Reference	31
10.2.1 Macro Definition Documentation	33
10.2.1.1 _WINSOCK_DEPRECATED_NO_WARNINGS	33
10.2.1.2 _XBOX_CONTROLLER_H_	33
10.2.1.3 BUFFER_LEN	33
10.2.1.4 GNUPLOT	33
10.2.1.5 USE_HAPTICS	33
10.2.1.6 USE_KINECT	33
10.2.1.7 USE_KINOVA	33
10.2.1.8 USE_TCP	33
10.2.1.9 WIN32_LEAN_AND_MEAN	33
10.2.1.10 WIN32_LEAN_AND_MEAN	33
10.2.1.11 WIN32_LEAN_AND_MEAN	33
10.3 expt_HoloLensCartesianTeleop.cpp File Reference	33
10.3.1 Macro Definition Documentation	34
10.3.1.1 INVERSE_CONTROLLER	34
10.3.1.2 LOAD_UHX_FROM_FILE	34
10.3.1.3 LOAD_XM_FROM_FILE	34
10.3.1.4 TEST_1	34
10.3.1.5 TEST_2	34
10.3.1.6 TRAIN	34
10.3.2 Variable Documentation	34
10.3.2.1 novintFalcon	34
10.4 firFilter.cpp File Reference	34
10.4.1 Macro Definition Documentation	35

10.4.1.1	<code>_WINSOCK_DEPRECATED_NO_WARNINGS</code>	35
10.5	HMIfunctions.cpp File Reference	35
10.5.1	Macro Definition Documentation	35
10.5.1.1	<code>_WINSOCK_DEPRECATED_NO_WARNINGS</code>	35
10.6	KinovaHMICartesianControl.cpp File Reference	35
10.6.1	Function Documentation	36
10.6.1.1	<code>main(int argc, char *argv[])</code>	36
10.6.2	Variable Documentation	36
10.6.2.1	<code>commandLayer_handle</code>	36
10.7	lutLinearInterp.cpp File Reference	36
10.7.1	Macro Definition Documentation	36
10.7.1.1	<code>_WINSOCK_DEPRECATED_NO_WARNINGS</code>	36
10.7.1.2	<code>WIN32_LEAN_AND_MEAN</code>	36
10.8	socketInConnection.cpp File Reference	36
10.8.1	Macro Definition Documentation	37
10.8.1.1	<code>_WINSOCK_DEPRECATED_NO_WARNINGS</code>	37
10.8.1.2	<code>HEADER_LENGTH</code>	37
10.9	socketInConnection.h File Reference	37
10.10	socketOutConnection.h File Reference	37
10.11	stdafx.cpp File Reference	38
10.12	stdafx.h File Reference	39
10.13	targetver.h File Reference	39

# 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.

- [Software License](#)
- [Installation Notes](#)

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

MIT License

Copyright (c) 2018 Rahul B. Warriar

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.



## 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-bit 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:

<a href="#">CSocketInConnection</a>	??
<a href="#">CSocketOutConnection</a>	??
<a href="#">CXBOXController</a>	??
<a href="#">Experiment</a>	??
<a href="#">FIRFilter</a>	??
<a href="#">kinectSkelTrack::KinectInfo</a>	??
<a href="#">kinectSkelTrack</a>	??
<a href="#">KinovaAPIFunctions</a>	??
<a href="#">NovintFalconHapticsDevice</a>	??
<a href="#">Timer</a>	??





## Chapter 8

# File Index

### 8.1 File List

Here is a list of all files with brief descriptions:

<a href="#">angularCommandControl.cpp</a>	??
<a href="#">CartesianControl.h</a>	??
<a href="#">expt_HoloLensCartesianTeleop.cpp</a>	??
<a href="#">firFilter.cpp</a>	??
<a href="#">HMIfunctions.cpp</a>	??
<a href="#">KinovaHMICartesianControl.cpp</a>	??
<a href="#">lutLinearInterp.cpp</a>	??
<a href="#">socketInConnection.cpp</a>	??
<a href="#">socketInConnection.h</a>	??
<a href="#">socketOutConnection.h</a>	??
<a href="#">stdafx.cpp</a>	??
<a href="#">stdafx.h</a>	??
<a href="#">targetver.h</a>	??



## 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_PARAM_SIZE], double OptimalzParam[OPTIMAL_Z_PARAM_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]`

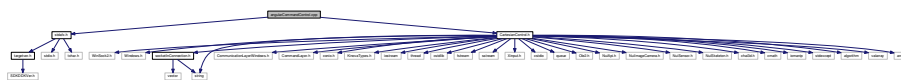
The documentation for this class was generated from the following file:

- [CartesianControl.h](#)



## File Documentation

```
#include "stdafx.h"
#include "CartesianControl.h"
Include dependency graph for angularCommandControl.cpp:
```



- #define \_WINSOCK\_DEPRECATED\_NO\_WARNINGS
- #define WIN32\_LEAN\_AND\_MEAN

```
10.1.1.1 #define _WINSOCK_DEPRECATED_NO_WARNINGS
```

## 10.2 CartesianControl.h File Reference

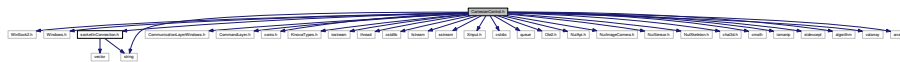
```
#include <WinSock2.h>
```

```

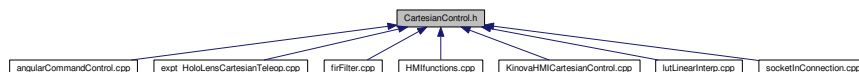
#include <Windows.h>
#include "socketInConnection.h"
#include "CommunicationLayerWindows.h"
#include "CommandLayer.h"
#include <conio.h>
#include "KinovaTypes.h"
#include <iostream>
#include <thread>
#include <cstdlib>
#include <fstream>
#include <string>
#include <sstream>
#include <Xinput.h>
#include <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](#)

- #define WINSOCKET\_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 GNUPLLOT false
- #define XBOX\_CONTROLLER\_H
- #define WIN32\_LEAN\_AND\_MEAN
- #define BUFFER\_LEN 96

## 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`

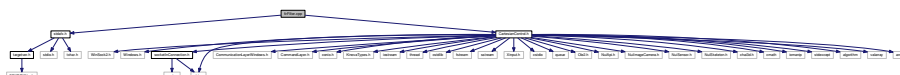
10.3.1.6 `#define TRAIN true`

### 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

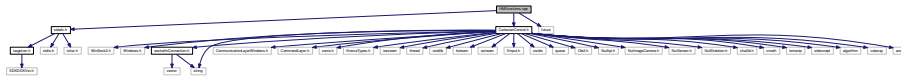
- `#define _WINSOCK_DEPRECATED_NO_WARNINGS`

### 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

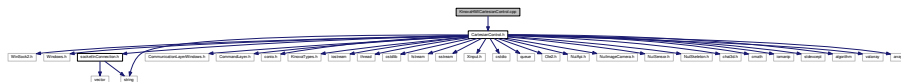
- `#define _WINSOCK_DEPRECATED_NO_WARNINGS`

### 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`



## 10.8.1 Macro Definition Documentation

10.8.1.1 `#define _WINSOCK_DEPRECATED_NO_WARNINGS`

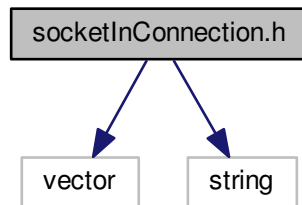
10.8.1.2 `#define HEADER_LENGTH 6`

## 10.9 socketInConnection.h File Reference

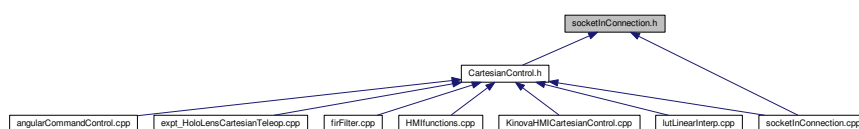
```
#include <vector>
```

```
#include <string>
```

Include dependency graph for socketInConnection.h:



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



## Classes

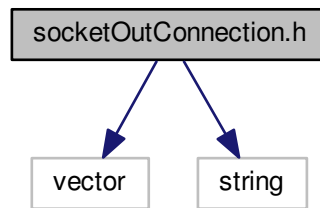
- class [CSocketInConnection](#)

## 10.10 socketOutConnection.h File Reference

```
#include <vector>
```

```
#include <string>
```

Include dependency graph for socketOutConnection.h:



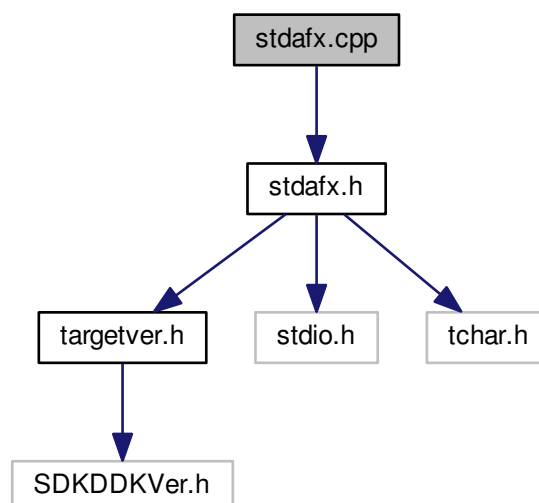
## Classes

- class [CSocketOutConnection](#)

## 10.11 stdafx.cpp File Reference

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

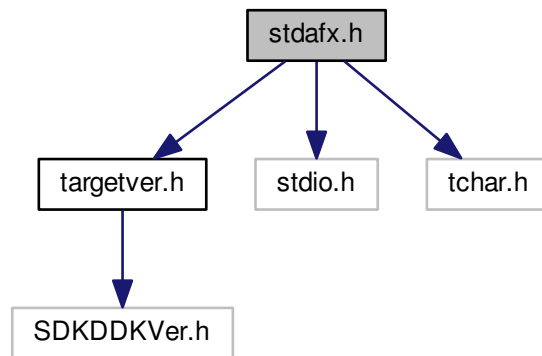
Include dependency graph for stdafx.cpp:



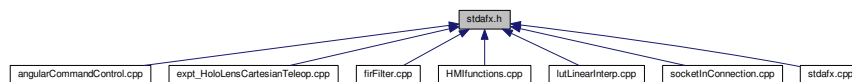


## 10.12 stdafx.h File Reference

```
#include "targetver.h"  
#include <stdio.h>  
#include <tchar.h>  
Include dependency graph for stdafx.h:
```

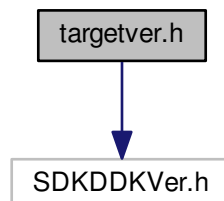


This graph shows which files directly or indirectly include this file:

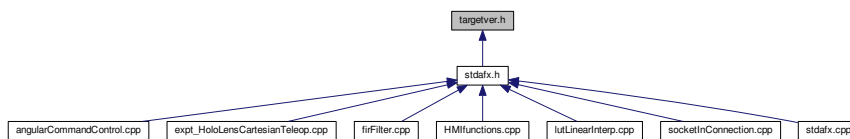


## 10.13 targetver.h File Reference

```
#include <SDKDDKVer.h>  
Include dependency graph for targetver.h:
```



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



# Index

- `_WINSOCK_DEPRECATED_NO_WARNINGS`
    - `angularCommandControl.cpp`, [31](#)
    - `CartesianControl.h`, [33](#)
    - `firFilter.cpp`, [35](#)
    - `HMIfunctions.cpp`, [35](#)
    - `lutLinearInterp.cpp`, [36](#)
    - `socketInConnection.cpp`, [37](#)
  - `_XBOX_CONTROLLER_H_`
    - `CartesianControl.h`, [33](#)
  - `_getTimeDiffInMs`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [19](#)
  - `_getTimeInMs`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [19](#)
  - `_headerByte1`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [20](#)
  - `_headerByte2`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [20](#)
  - `_maxPacketSize`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [20](#)
  - `_receiveSimplePacket`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [19](#)
  - `_sendSimplePacket`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [19](#)
  - `_socketClient`
    - `CSocketInConnection`, [18](#)
  - `_socketConn`
    - `CSocketOutConnection`, [20](#)
  - `_socketConnectWasOk`
    - `CSocketInConnection`, [18](#)
  - `_socketConnectedMachineIP`
    - `CSocketInConnection`, [18](#)
  - `_socketConnectionAddress`
    - `CSocketOutConnection`, [20](#)
  - `_socketConnectionPort`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [20](#)
  - `_socketLocal`
    - `CSocketInConnection`, [18](#)
  - `_socketServer`
    - `CSocketInConnection`, [18](#)
    - `CSocketOutConnection`, [20](#)
  - `_socketTheSet`
- `CSocketInConnection`, [18](#)
- `~CSocketInConnection`
  - `CSocketInConnection`, [18](#)
- `~CSocketOutConnection`
  - `CSocketOutConnection`, [19](#)
- `angularCommandControl.cpp`, [31](#)
  - `_WINSOCK_DEPRECATED_NO_WARNINGS`, [31](#)
  - `WIN32_LEAN_AND_MEAN`, [31](#)
- `angularVelocity`
  - `NovintFalconHapticsDevice`, [29](#)
- `BUFFER_LEN`
  - `CartesianControl.h`, [33](#)
- `button0_state`
  - `NovintFalconHapticsDevice`, [29](#)
- `button2_state`
  - `NovintFalconHapticsDevice`, [29](#)
- `CSocketInConnection`, [17](#)
  - `_getTimeDiffInMs`, [18](#)
  - `_getTimeInMs`, [18](#)
  - `_headerByte1`, [18](#)
  - `_headerByte2`, [18](#)
  - `_maxPacketSize`, [18](#)
  - `_receiveSimplePacket`, [18](#)
  - `_sendSimplePacket`, [18](#)
  - `_socketClient`, [18](#)
  - `_socketConnectWasOk`, [18](#)
  - `_socketConnectedMachineIP`, [18](#)
  - `_socketConnectionPort`, [18](#)
  - `_socketLocal`, [18](#)
  - `_socketServer`, [18](#)
  - `_socketTheSet`, [18](#)
  - `~CSocketInConnection`, [18](#)
  - `CSocketInConnection`, [18](#)
  - `connectToClient`, [18](#)
  - `getConnectedMachineIP`, [18](#)
  - `receiveData`, [18](#)
  - `replyToReceivedData`, [18](#)
- `CSocketOutConnection`, [19](#)
  - `_getTimeDiffInMs`, [19](#)
  - `_getTimeInMs`, [19](#)
  - `_headerByte1`, [20](#)
  - `_headerByte2`, [20](#)
  - `_maxPacketSize`, [20](#)
  - `_receiveSimplePacket`, [19](#)
  - `_sendSimplePacket`, [19](#)
  - `_socketConn`, [20](#)
  - `_socketConnectionAddress`, [20](#)

- \_socketConnectionPort, 20
  - \_socketServer, 20
  - ~CSocketOutConnection, 19
  - CSocketOutConnection, 19
  - connectToServer, 20
  - receiveReplyData, 20
  - sendData, 20
- CXBOXController, 20
  - CXBOXController, 21
  - GetState, 21
  - IsConnected, 21
  - Vibrate, 21
- CartesianControl.h, 31
  - \_WINSOCK\_DEPRECATED\_NO\_WARNINGS, 33
  - \_XBOX\_CONTROLLER\_H\_, 33
  - BUFFER\_LEN, 33
  - GNUPLOT, 33
  - USE\_HAPTICS, 33
  - USE\_KINECT, 33
  - USE\_KINOVA, 33
  - USE\_TCP, 33
  - WIN32\_LEAN\_AND\_MEAN, 33
- coeffs
  - FIRFilter, 24
- commandLayer\_handle
  - KinovaHMICartesianControl.cpp, 36
- connectToClient
  - CSocketInConnection, 18
- connectToServer
  - CSocketOutConnection, 20
- cutoff\_freq\_hz
  - FIRFilter, 24
- depthStream
  - kinectSkelTrack, 26
- desiredPosition
  - NovintFalconHapticsDevice, 29
- desiredRotation
  - NovintFalconHapticsDevice, 29
- elapsedTime
  - Timer, 30
- Experiment, 21
  - fs\_qd\_lut, 23
  - fs\_uc\_lut, 23
  - HoloLensCartesianTeleop, 22
  - interp\_lut, 22
  - Kp, 23
  - LUT, 23
  - len\_LUT, 23
  - load\_LUT1D, 22
  - MoveEndEffectorPos, 22
  - MovetoStartPos, 23
  - scale\_LUT, 23
  - T, 23
  - T\_TF, 23
  - T\_calib, 23
  - Ts, 23
- expt\_HoloLensCartesianTeleop.cpp, 33
- INVERSE\_CONTROLLER, 34
- LOAD\_UHX\_FROM\_FILE, 34
- LOAD\_XM\_FROM\_FILE, 34
- novintFalcon, 34
- TEST\_1, 34
- TEST\_2, 34
- TRAIN, 34
- FIRFilter, 23
  - coeffs, 24
  - cutoff\_freq\_hz, 24
  - filename, 25
  - firFloat, 24
  - firFloatInit, 24
  - insamp, 25
  - length, 25
  - output, 25
- filename
  - FIRFilter, 25
- firFilter.cpp, 34
  - \_WINSOCK\_DEPRECATED\_NO\_WARNINGS, 35
- firFloat
  - FIRFilter, 24
- firFloatInit
  - FIRFilter, 24
- fs\_qd\_lut
  - Experiment, 23
- fs\_uc\_lut
  - Experiment, 23
- GNUPLOT
  - CartesianControl.h, 33
- getConnectedMachineIP
  - CSocketInConnection, 18
- getKinectData
  - kinectSkelTrack, 26
- getSkeletalData
  - kinectSkelTrack, 26
- GetState
  - CXBOXController, 21
- HEADER\_LENGTH
  - socketInConnection.cpp, 37
- HMIfunctions.cpp, 35
  - \_WINSOCK\_DEPRECATED\_NO\_WARNINGS, 35
- handPosition
  - kinectSkelTrack::KinectInfo, 25
- handler
  - NovintFalconHapticsDevice, 29
- hapticDevice
  - NovintFalconHapticsDevice, 29
- hapticDevicePosition
  - NovintFalconHapticsDevice, 30
- hapticsThread
  - NovintFalconHapticsDevice, 30
- height
  - kinectSkelTrack, 26
- HoloLensCartesianTeleop
  - Experiment, 22

- INVERSE\_CONTROLLER
  - expt\_HoloLensCartesianTeleop.cpp, [34](#)
- initKinect
  - kinectSkelTrack, [26](#)
- InitializeHapticsDevice
  - NovintFalconHapticsDevice, [29](#)
- insamp
  - FIRFilter, [25](#)
- interp\_lut
  - Experiment, [22](#)
- IsConnected
  - CXBOXController, [21](#)
- isRunning
  - NovintFalconHapticsDevice, [30](#)
- isTimeout
  - Timer, [30](#)
- kinectSkelTrack, [26](#)
  - depthStream, [26](#)
  - getKinectData, [26](#)
  - getSkeletalData, [26](#)
  - height, [26](#)
  - initKinect, [26](#)
  - sensor, [26](#)
  - skeletonPosition, [26](#)
  - width, [26](#)
- kinectSkelTrack::KinectInfo, [25](#)
  - handPosition, [25](#)
  - startSignal, [25](#)
  - userFound, [25](#)
- KinovaAPIFunctions, [27](#)
  - MyCloseAPI, [27](#)
  - MyGetActualTrajectoryInfo, [27](#)
  - MyGetAngularPosition, [27](#)
  - MyGetCartesianCommand, [27](#)
  - MyGetCartesianPosition, [27](#)
  - MyGetClientConfigurations, [27](#)
  - MyGetDevices, [27](#)
  - MyGetGlobalTrajectoryInfo, [27](#)
  - MyGetGripperStatus, [27](#)
  - MyInitAPI, [28](#)
  - MyInitFingers, [28](#)
  - MyMoveHome, [28](#)
  - MyRunGravityZEstimationSequence, [28](#)
  - MySendBasicTrajectory, [28](#)
  - MySendJoystickCommand, [28](#)
  - MySetActiveDevice, [28](#)
  - MySetActuatorPID, [28](#)
  - MySetClientConfigurations, [28](#)
  - MySetGravityOptimalZParam, [28](#)
  - MySetGravityType, [28](#)
  - MyStartControlAPI, [28](#)
  - MyStartForceControl, [28](#)
  - MyStopForceControl, [28](#)
- KinovaHMICartesianControl.cpp, [35](#)
  - commandLayer\_handle, [36](#)
  - main, [36](#)
- Kp
  - Experiment, [23](#)
- LOAD\_UHX\_FROM\_FILE
  - expt\_HoloLensCartesianTeleop.cpp, [34](#)
- LOAD\_XM\_FROM\_FILE
  - expt\_HoloLensCartesianTeleop.cpp, [34](#)
- LUT
  - Experiment, [23](#)
- len\_LUT
  - Experiment, [23](#)
- length
  - FIRFilter, [25](#)
- linearVelocity
  - NovintFalconHapticsDevice, [30](#)
- load\_LUT1D
  - Experiment, [22](#)
- lutLinearInterp.cpp, [36](#)
  - \_WINSOCK\_DEPRECATED\_NO\_WARNINGS, [36](#)
  - WIN32\_LEAN\_AND\_MEAN, [36](#)
- main
  - KinovaHMICartesianControl.cpp, [36](#)
- maxforce
  - NovintFalconHapticsDevice, [30](#)
- MoveEndEffectorPos
  - Experiment, [22](#)
- MovetoStartPos
  - Experiment, [23](#)
- MyCloseAPI
  - KinovaAPIFunctions, [27](#)
- MyGetActualTrajectoryInfo
  - KinovaAPIFunctions, [27](#)
- MyGetAngularPosition
  - KinovaAPIFunctions, [27](#)
- MyGetCartesianCommand
  - KinovaAPIFunctions, [27](#)
- MyGetCartesianPosition
  - KinovaAPIFunctions, [27](#)
- MyGetClientConfigurations
  - KinovaAPIFunctions, [27](#)
- MyGetDevices
  - KinovaAPIFunctions, [27](#)
- MyGetGlobalTrajectoryInfo
  - KinovaAPIFunctions, [27](#)
- MyGetGripperStatus
  - KinovaAPIFunctions, [27](#)
- MyInitAPI
  - KinovaAPIFunctions, [28](#)
- MyInitFingers
  - KinovaAPIFunctions, [28](#)
- MyMoveHome
  - KinovaAPIFunctions, [28](#)
- MyRunGravityZEstimationSequence
  - KinovaAPIFunctions, [28](#)
- MySendBasicTrajectory
  - KinovaAPIFunctions, [28](#)
- MySendJoystickCommand
  - KinovaAPIFunctions, [28](#)
- MySetActiveDevice
  - KinovaAPIFunctions, [28](#)
- MySetActuatorPID

- KinovaAPIFunctions, 28
- MySetClientConfigurations
  - KinovaAPIFunctions, 28
- MySetGravityOptimalZParam
  - KinovaAPIFunctions, 28
- MySetGravityType
  - KinovaAPIFunctions, 28
- MyStartControlAPI
  - KinovaAPIFunctions, 28
- MyStartForceControl
  - KinovaAPIFunctions, 28
- MyStopForceControl
  - KinovaAPIFunctions, 28
- novintFalcon
  - expt\_HoloLensCartesianTeleop.cpp, 34
- NovintFalconHapticsDevice, 28
  - angularVelocity, 29
  - button0\_state, 29
  - button2\_state, 29
  - desiredPosition, 29
  - desiredRotation, 29
  - handler, 29
  - hapticDevice, 29
  - hapticDevicePosition, 30
  - hapticsThread, 30
  - InitializeHapticsDevice, 29
  - isRunning, 30
  - linearVelocity, 30
  - maxforce, 30
  - position, 30
  - ResetIC, 29
  - rotation, 30
  - UpdateHaptics, 29
  - useDamping, 30
  - useForceField, 30
- output
  - FIRFilter, 25
- position
  - NovintFalconHapticsDevice, 30
- receiveData
  - CSocketInConnection, 18
- receiveReplyData
  - CSocketOutConnection, 20
- replyToReceivedData
  - CSocketInConnection, 18
- ResetIC
  - NovintFalconHapticsDevice, 29
- rotation
  - NovintFalconHapticsDevice, 30
- scale\_LUT
  - Experiment, 23
- sendData
  - CSocketOutConnection, 20
- sensor
  - kinectSkelTrack, 26
  - skeletonPosition
    - kinectSkelTrack, 26
  - socketInConnection.cpp, 36
    - \_WINSOCK\_DEPRECATED\_NO\_WARNINGS, 37
    - HEADER\_LENGTH, 37
  - socketInConnection.h, 37
  - socketOutConnection.h, 37
  - start
    - Timer, 30
  - startSignal
    - kinectSkelTrack::KinectInfo, 25
  - stdafx.cpp, 38
  - stdafx.h, 39
- T
  - Experiment, 23
- T\_TF
  - Experiment, 23
- T\_calib
  - Experiment, 23
- TEST\_1
  - expt\_HoloLensCartesianTeleop.cpp, 34
- TEST\_2
  - expt\_HoloLensCartesianTeleop.cpp, 34
- TRAIN
  - expt\_HoloLensCartesianTeleop.cpp, 34
- targetver.h, 39
- Timer, 30
  - elapsedTime, 30
  - isTimeout, 30
  - start, 30
- Ts
  - Experiment, 23
- USE\_HAPTICS
  - CartesianControl.h, 33
- USE\_KINECT
  - CartesianControl.h, 33
- USE\_KINOVA
  - CartesianControl.h, 33
- USE\_TCP
  - CartesianControl.h, 33
- UpdateHaptics
  - NovintFalconHapticsDevice, 29
- useDamping
  - NovintFalconHapticsDevice, 30
- useForceField
  - NovintFalconHapticsDevice, 30
- userFound
  - kinectSkelTrack::KinectInfo, 25
- Vibrate
  - CXBOXController, 21
- WIN32\_LEAN\_AND\_MEAN
  - angularCommandControl.cpp, 31
  - CartesianControl.h, 33
  - lutLinearInterp.cpp, 36

width

kinectSkelTrack, [26](#)