Stewart Controller

1.0

Generated by Doxygen 1.8.17

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

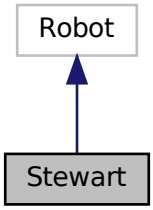
Class Documentation

4.1 Stewart Class Reference

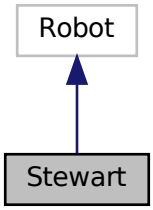
Stewart Controller Class This class takes care of interfacing with Webots control API and implementing kinematics and trajectory plan for the Stewart Platform.

`#include <stewart_controller.hpp>`

Inheritance diagram for Stewart:



Collaboration diagram for Stewart:



Public Member Functions

- Stewart (int argc, char **argv, std::string node_name)

  Construct a new Stewart object It initializes the ROS node, the Webots Interface, reads the YAML config file, ROS params and initialise the attributes needed for the project.
- double get_force_feedback (int id)

  Get the force feedback object.
- void set_piston_pos (int id, double pos)

  Set the piston extension.
- std::tuple< VectorXd, MatrixXd > inverse_kinematics (VectorXd setpoint)

  Method for computing the inverse kinematics of the defined Stewart Platform.
- VectorXd forward_kinematics (VectorXd pose_guess, VectorXd joint_pos)

  Method for computing the forward kinematics of the defined Stewart Platform using Newton-Raphson algorithm.
- MatrixXd inverse_jacobian (VectorXd base_pose)

  Method for computing the inverse jacobian matrix (map from E-E linear and angular velocities to Joints Velocities)
- void reach_setpoint_trapz ()

  routine to reach a desired setpoint using a trapezoidal velocity trajectory
- void reach_setpoint ()

  routine to reach a setpoint using inverse kinematics
- void reach_setpoint (VectorXd setpoint)

  routine to reach a setpoint using inverse kinematics
- VectorXd get_joints_pos ()

  Get the joints position.
- VectorXd get_base_pose ()

  Get the movng base pose groundtruth pose.
- VectorXd get_base_vel ()

  Get the moving base groundtruth velocity.
- void estimate_base_pose ()

  Routine to get estimation of base pose position using Forward Kinematics.
- void set_piston_vel (int id, double vel)

  Set the piston #ID velocity.
- void set_target_vel ()

  Routine to set the velocity of the moving base to the desired setpoint velocity (modified by ROS callback)
- void set_target_vel (Eigen::VectorXd target)

  Routine to set the velocity of the moving base to the desired setpoint velocity (parameter)
- int get_mode ()

  Get the actual controlling mode.

Private Member Functions

- void enable_devices ()

  Routine to enable Webots sensors.
- void init_vectors ()

  Routine to initialize Eigen Vectors.
- VectorXd convert_6d_to_7d (VectorXd euler_pos)

  Method to convert a 6D position + euler orientation vector into a 7D position + quaternion orientation vector.
- VectorXd convert_7d_to_6d (VectorXd quat_pos)

  Method to convert a 7D position + quaternion orientation vector into a 6D position + euler orientation vector.
- MatrixXd inv_J_1 (MatrixXd n, Quaterniond q)

  method to compute J_1^{-1} according to [1]

- MatrixXd inv_J_2 (Quaterniond q)

  method to compute J_2^{-1} according to [1]
- double trapezoidal_target (double qi, double qf, double time, bool angular)

  Method to compute geometrically the trapezoidal velocity trajectory.
- Eigen::Matrix4d skew_matrix (Vector3d v)

  Method to compute the skew matrix from a 3D vector.
- bool trapezoidal_trajectory (VectorXd qi, VectorXd qf, double time)

  Method that takes care of integrating the desired velocities and set target position in trapezoidal velocity trajectory control.
- void retrieve_params (ros::NodeHandle &nodeHandle_)

  Routine to retrieve roslaunch params.
- void init_pubs_subs (ros::NodeHandle &nodeHandle_)

  routine to initialise ROS publisher and subcribers
- void joy_callback (const sensor_msgs::Joy &msg)

  Joystick callback.
- void setpoint_callback (const geometry_msgs::Pose &msg)

  Pose setpoint callback (for IK pose control)
- void setpoint_trapz_callback (const geometry_msgs::Pose &msg)

  Pose setpoint callback (for trapezoidal velocity pose control)
- void setpoint_vel_callback (const geometry_msgs::Twist &msg)

  Twist setpoint callback (for twist control)
- void change_acc_callback (const std_msgs::Float32 &msg)

  Callback to change acc value in trapezoidal control.
- void change_vel_callback (const std_msgs::Float32 &msg)

  Callback to change vel value in trapezoidal control.
- void change_mod_callback (const std_msgs::Int32 &msg)

  Callback to change mode value.

Private Attributes

- std::string config_file_
- std::vector< Motor * > pistons_
- std::vector< PositionSensor * > pistons_pos_
- GPS * gps_upp_plat
- InertialUnit * att_upp_plat
- Gyro * ang_vel_upp_plat
- Accelerometer * acc_upp_plat
- int mode_
- YAML::Node config_stewart
- MatrixXd B
- MatrixXd A
- VectorXd setpoint_
- VectorXd setpoint_trapz_
- VectorXd base_pose_
- VectorXd base_pose_gt_
- VectorXd setpoint_vel
- VectorXd base_vel_
- VectorXd qi_
- VectorXd qf_
- VectorXd target_
- double old_time_
- double time_

- bool init_trapz_ = false
- bool trapz_initialization = false
- double trapz_acc_
- double trapz_max_vel_
- ros::Publisher pose_cmd_pub_
- ros::Publisher pose_pub_
- ros::Publisher pose_vel_pub_
- ros::Subscriber setpoint_sub_
- ros::Subscriber joy_sub_
- ros::Subscriber trapz_setpoint_sub_
- ros::Subscriber setpoint_vel_sub_
- ros::Subscriber change_acc_sub_
- ros::Subscriber change_vel_sub_
- ros::Subscriber change_mod_sub_

4.1.1 Detailed Description

Stewart Controller Class This class takes care of interfacing with Webots control API and implementing kinematics and trajectory plan for the Stewart Platform.

Kinematics are implemented according to "Kinematic and dynamic analysis of Stewart platform-based machine tool structures" by Khalifa Harib and Krishnaswamy Srinivasan [1]

Author

Fabio Polisano

Definition at line 36 of file stewart_controller.hpp.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Stewart()

```
Stewart::Stewart (
            int argc,
            char ** argv,
            std::string node_name )
```

Construct a new Stewart object It initializes the ROS node, the Webots Interface, reads the YAML config file, ROS params and initialise the attributes needed for the project.

Parameters

argc	number of strings pointed by argv
argv	argument vector
node_name	name of ROS node

4.1.3 Member Function Documentation

4.1.3.1 change_acc_callback()

```
void Stewart::change_acc_callback (
            const std_msgs::Float32 & msg )  [private]
```

Callback to change acc value in trapezoidal control.

Parameters

msg	

4.1.3.2 change_mod_callback()

```
void Stewart::change_mod_callback (
            const std_msgs::Int32 & msg )  [private]
```

Callback to change mode value.

Parameters

msg	

4.1.3.3 change_vel_callback()

```
void Stewart::change_vel_callback (
            const std_msgs::Float32 & msg )  [private]
```

Callback to change vel value in trapezoidal control.

Parameters

msg	

4.1.3.4 convert_6d_to_7d()

```
VectorXd Stewart::convert_6d_to_7d (
            VectorXd euler_pos )  [private]
```

Method to convert a 6D position + euler orientation vector into a 7D position + quaternion orientation vector.

Parameters

euler_pos	6D position + euler orientation vector

Returns

VectorXd 7D position + quaternion orientation vector

4.1.3.5 convert_7d_to_6d()

```
VectorXd Stewart::convert_7d_to_6d (
            VectorXd quat_pos )  [private]
```

Method to convert a 7D position + quaternion orientation vector into a 6D position + euler orientation vector.

Parameters

quat_pos	7D position + quaternion orientation vector

Returns

VectorXd 6D position + euler orientation vector

4.1.3.6 enable_devices()

```
void Stewart::enable_devices ( )  [private]
```

Routine to enable Webots sensors.

4.1.3.7 estimate_base_pose()

```
void Stewart::estimate_base_pose ( )
```

Routine to get estimation of base pose position using Forward Kinematics.

4.1.3.8 forward_kinematics()

```
VectorXd Stewart::forward_kinematics (
            VectorXd pose_guess,
            VectorXd joint_pos )
```

Method for computing the forward kinematics of the defined Stewart Platform using Newton-Raphson algorithm.

Parameters

pose_guess	a 7D vector containing an initial guess of the E-E pose
joint_pos	a (NUM_PISTONS)D vector containing the configuration of the Joints

Returns

VectorXd a 7D vector containing the E-E pose estimatio

4.1.3.9 get_base_pose()

```
VectorXd Stewart::get_base_pose ( )
```

Get the movng base pose groundtruth pose.

Returns

VectorXd 7D vector containing moving base position and attitude

4.1.3.10 get_base_vel()

```
VectorXd Stewart::get_base_vel ( )
```

Get the moving base groundtruth velocity.

Returns

VectorXd 6D vector containing moving base linear and angular velocity

4.1.3.11 get_force_feedback()

```
double Stewart::get_force_feedback (
            int id )
```

Get the force feedback object.

Parameters

id	ID of the motor

Returns

double Force feedback from Motor #ID

4.1.3.12 get_joints_pos()

```
VectorXd Stewart::get_joints_pos ( )
```

Get the joints position.

Returns

VectorXd (NUM_PISTON)D vector containing the joints position

4.1.3.13 get_mode()

```
int Stewart::get_mode ( )
```

Get the actual controlling mode.

Returns

int 1 Trapezoidal 2 Twist 3 Position using IK 4 FK estimation

4.1.3.14 init_pubs_subs()

```
void Stewart::init_pubs_subs (
            ros::NodeHandle & nodeHandle_ )  [private]
```

routine to initialise ROS publisher and subcribers

Parameters

*node↩ Handle_*	

4.1.3.15 init_vectors()

```
void Stewart::init_vectors ( )  [private]
```

Routine to initialize Eigen Vectors.

4.1.3.16 inv_J_1()

```
MatrixXd Stewart::inv_J_1 (
            MatrixXd n,
            Quaterniond q )  [private]
```

method to compute J_1^{-1} according to [1]

Parameters

n	6x3 Matrix containing the direction vectors of tje joints
q	quaternion containing actual orientation of the moving base

Returns

MatrixXd J_1^-{1}

4.1.3.17 inv_J_2()

```
MatrixXd Stewart::inv_J_2 (
            Quaterniond q )  [private]
```

method to compute J_2^{-1} according to [1]

Parameters

q	quaternion containing actual orientation of the moving base

Returns

MatrixXd J_2^-{1}

4.1.3.18 inverse_jacobian()

```
MatrixXd Stewart::inverse_jacobian (
            VectorXd base_pose )
```

Method for computing the inverse jacobian matrix (map from E-E linear and angular velocities to Joints Velocities)

Parameters

base_pose	a 7D vector containing the E-E pose

Returns

MatrixXd a 6x7 matrix

4.1.3.19 inverse_kinematics()

```
std::tuple<VectorXd, MatrixXd> Stewart::inverse_kinematics (
            VectorXd setpoint )
```

Method for computing the inverse kinematics of the defined Stewart Platform.

Parameters

setpoint	a 7D vector containg desired position (x,y,z) and orientation quaternion (w,x,y,z)

Returns

std::tuple<VectorXd, MatrixXd> a tuple containing a (#NUM_PISTONS)D vector containing the joints length and a matrix (#NUM_PISTONSx3)D containing the direction vectors of the pistons (needed for the jacobian computation)

4.1.3.20 joy_callback()

```
void Stewart::joy_callback (
            const sensor_msgs::Joy & msg )  [private]
```

Joystick callback.

Parameters

msg	

4.1.3.21 reach_setpoint() [1/2]

```
void Stewart::reach_setpoint ( )
```

routine to reach a setpoint using inverse kinematics

4.1.3.22 reach_setpoint() [2/2]

```
void Stewart::reach_setpoint (
            VectorXd setpoint )
```

routine to reach a setpoint using inverse kinematics

Parameters

setpoint	7D vector desired setpoint

4.1.3.23 reach_setpoint_trapz()

```
void Stewart::reach_setpoint_trapz ( )
```

routine to reach a desired setpoint using a trapezoidal velocity trajectory

4.1.3.24 retrieve_params()

```
void Stewart::retrieve_params (
            ros::NodeHandle & nodeHandle_ )  [private]
```

Routine to retrieve roslaunch params.

Parameters

*node↩ Handle_*	

4.1.3.25 set_piston_pos()

```
void Stewart::set_piston_pos (
            int id,
            double pos )
```

Set the piston extension.

Parameters

id	ID of the motor
pos	Desired position

4.1.3.26 set_piston_vel()

```
void Stewart::set_piston_vel (
            int id,
            double vel )
```

Set the piston #ID velocity.

Parameters

id	ID of the motor
vel	Desired velocity of motor #ID

4.1.3.27 set_target_vel() [1/2]

```
void Stewart::set_target_vel ( )
```

Routine to set the velocity of the moving base to the desired setpoint velocity (modified by ROS callback)

4.1.3.28 set_target_vel() [2/2]

```
void Stewart::set_target_vel (
            Eigen::VectorXd target )
```

Routine to set the velocity of the moving base to the desired setpoint velocity (parameter)

Parameters

target	6D vector containing desired moving base velocity

4.1.3.29 setpoint_callback()

```
void Stewart::setpoint_callback (
            const geometry_msgs::Pose & msg )  [private]
```

Pose setpoint callback (for IK pose control)

Parameters

msg	

4.1.3.30 setpoint_trapz_callback()

```
void Stewart::setpoint_trapz_callback (
            const geometry_msgs::Pose & msg )  [private]
```

Pose setpoint callback (for trapezoidal velocity pose control)

Parameters

msg	

4.1.3.31 setpoint_vel_callback()

```
void Stewart::setpoint_vel_callback (
            const geometry_msgs::Twist & msg )  [private]
```

Twist setpoint callback (for twist control)

Parameters

msg	

4.1.3.32 skew_matrix()

```
Eigen::Matrix4d Stewart::skew_matrix (
            Vector3d v )  [private]
```

Method to compute the skew matrix from a 3D vector.

Parameters

v	3D Vector

Returns

Eigen::Matrix4d Skew Matrix of v

4.1.3.33 trapezoidal_target()

```
double Stewart::trapezoidal_target (
            double qi,
```

```
            double qf,
            double time,
            bool angular )  [private]
```

Method to compute geometrically the trapezoidal velocity trajectory.

Parameters

qi	initial pose of the moving base
qf	desired final pose of the moving base
time	time since the movement started
angular	if the function is computing the trajectory for angular variable

Returns

double target velocity

4.1.3.34 trapezoidal_trajectory()

```
bool Stewart::trapezoidal_trajectory (
            VectorXd qi,
            VectorXd qf,
            double time )  [private]
```

Method that takes care of integrating the desired velocities and set target position in trapezoidal velocity trajectory control.

Parameters

qi	initial pose
qf	final pose
time	time since the operation started

Returns

true

false

4.1.4 Member Data Documentation

4.1.4.1 A

```
MatrixXd Stewart::A  [private]
```

Definition at line 202 of file stewart_controller.hpp.

4.1.4.2 acc_upp_plat

```
Accelerometer* Stewart::acc_upp_plat  [private]
```

Definition at line 194 of file stewart_controller.hpp.

4.1.4.3 ang_vel_upp_plat

```
Gyro* Stewart::ang_vel_upp_plat  [private]
```

Definition at line 193 of file stewart_controller.hpp.

4.1.4.4 att_upp_plat

```
InertialUnit* Stewart::att_upp_plat  [private]
```

Definition at line 192 of file stewart_controller.hpp.

4.1.4.5 B

```
MatrixXd Stewart::B  [private]
```

Definition at line 201 of file stewart_controller.hpp.

4.1.4.6 base_pose_

```
VectorXd Stewart::base_pose_  [private]
```

Definition at line 206 of file stewart_controller.hpp.

4.1.4.7 base_pose_gt_

```
VectorXd Stewart::base_pose_gt_  [private]
```

Definition at line 207 of file stewart_controller.hpp.

4.1.4.8 base_vel_

```
VectorXd Stewart::base_vel_  [private]
```

Definition at line 210 of file stewart_controller.hpp.

4.1.4.9 change_acc_sub_

```
ros::Subscriber Stewart::change_acc_sub_  [private]
```

Definition at line 232 of file stewart_controller.hpp.

4.1.4.10 change_mod_sub_

```
ros::Subscriber Stewart::change_mod_sub_  [private]
```

Definition at line 234 of file stewart_controller.hpp.

4.1.4.11 change_vel_sub_

```
ros::Subscriber Stewart::change_vel_sub_  [private]
```

Definition at line 233 of file stewart_controller.hpp.

4.1.4.12 config_file_

```
std::string Stewart::config_file_  [private]
```

Definition at line 186 of file stewart_controller.hpp.

4.1.4.13 config_stewart

```
YAML::Node Stewart::config_stewart  [private]
```

Definition at line 199 of file stewart_controller.hpp.

4.1.4.14 gps_upp_plat

```
GPS* Stewart::gps_upp_plat  [private]
```

Definition at line 191 of file stewart_controller.hpp.

4.1.4.15 init_trapz_

```
bool Stewart::init_trapz_ = false  [private]
```

Definition at line 217 of file stewart_controller.hpp.

4.1.4.16 joy_sub_

```
ros::Subscriber Stewart::joy_sub_  [private]
```

Definition at line 229 of file stewart_controller.hpp.

4.1.4.17 mode_

```
int Stewart::mode_  [private]
```

Definition at line 197 of file stewart_controller.hpp.

4.1.4.18 old_time_

```
double Stewart::old_time_  [private]
```

Definition at line 214 of file stewart_controller.hpp.

4.1.4.19 pistons_

```
std::vector<Motor*> Stewart::pistons_  [private]
```

Definition at line 188 of file stewart_controller.hpp.

4.1.4.20 pistons_pos_

```
std::vector<PositionSensor*> Stewart::pistons_pos_  [private]
```

Definition at line 189 of file stewart_controller.hpp.

4.1.4.21 pose_cmd_pub_

```
ros::Publisher Stewart::pose_cmd_pub_  [private]
```

Definition at line 224 of file stewart_controller.hpp.

4.1.4.22 pose_pub_

```
ros::Publisher Stewart::pose_pub_  [private]
```

Definition at line 225 of file stewart_controller.hpp.

4.1.4.23 pose_vel_pub_

```
ros::Publisher Stewart::pose_vel_pub_  [private]
```

Definition at line 226 of file stewart_controller.hpp.

4.1.4.24 qf_

```
VectorXd Stewart::qf_  [private]
```

Definition at line 211 of file stewart_controller.hpp.

4.1.4.25 qi_

```
VectorXd Stewart::qi_  [private]
```

Definition at line 211 of file stewart_controller.hpp.

4.1.4.26 setpoint_

```
VectorXd Stewart::setpoint_  [private]
```

Definition at line 204 of file stewart_controller.hpp.

4.1.4.27 setpoint_sub_

```
ros::Subscriber Stewart::setpoint_sub_  [private]
```

Definition at line 228 of file stewart_controller.hpp.

4.1.4.28 setpoint_trapz_

```
VectorXd Stewart::setpoint_trapz_  [private]
```

Definition at line 205 of file stewart_controller.hpp.

4.1.4.29 setpoint_vel

```
VectorXd Stewart::setpoint_vel  [private]
```

Definition at line 208 of file stewart_controller.hpp.

4.1.4.30 setpoint_vel_sub_

```
ros::Subscriber Stewart::setpoint_vel_sub_  [private]
```

Definition at line 231 of file stewart_controller.hpp.

4.1.4.31 target_

```
VectorXd Stewart::target_  [private]
```

Definition at line 212 of file stewart_controller.hpp.

4.1.4.32 time_

```
double Stewart::time_  [private]
```

Definition at line 215 of file stewart_controller.hpp.

4.1.4.33 trapz_acc_

```
double Stewart::trapz_acc_  [private]
```

Definition at line 220 of file stewart_controller.hpp.

4.1.4.34 trapz_initialization

```
bool Stewart::trapz_initialization = false  [private]
```

Definition at line 218 of file stewart_controller.hpp.

4.1.4.35 trapz_max_vel_

```
double Stewart::trapz_max_vel_  [private]
```

Definition at line 221 of file stewart_controller.hpp.

4.1.4.36 trapz_setpoint_sub_

```
ros::Subscriber Stewart::trapz_setpoint_sub_  [private]
```

Definition at line 230 of file stewart_controller.hpp.

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

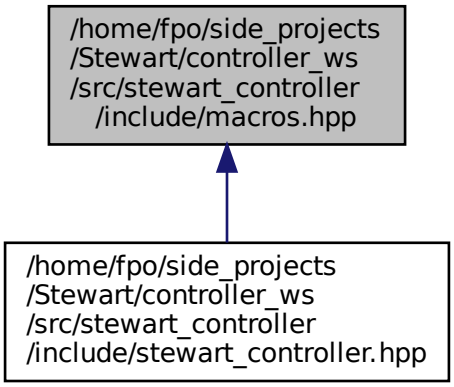
- /home/fpo/side_projects/Stewart/controller_ws/src/stewart_controller/include/stewart_controller.hpp

Chapter 5

File Documentation

5.1 /home/fpo/side_projects/Stewart/controller_ws/src/stewart_↩ controller/include/macros.hpp File Reference

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



Macros

- #define NUM_PISTONS 6

  Number of Pistons.
- #define BASE_Z 2.733406

  initial Z coordinate of moving base

5.1.1 Macro Definition Documentation

5.1.1.1 BASE_Z

```
#define BASE_Z 2.733406
```

initial Z coordinate of moving base

Definition at line 13 of file macros.hpp.

5.1.1.2 NUM_PISTONS

```
#define NUM_PISTONS 6
```

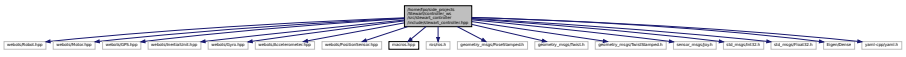
Number of Pistons.

Definition at line 7 of file macros.hpp.

5.2 /home/fpo/side_projects/Stewart/controller_ws/src/stewart_↩ controller/include/stewart_controller.hpp File Reference

```
#include <webots/Robot.hpp>
#include <webots/Motor.hpp>
#include <webots/GPS.hpp>
#include <webots/InertialUnit.hpp>
#include <webots/Gyro.hpp>
#include <webots/Accelerometer.hpp>
#include <webots/PositionSensor.hpp>
#include <macros.hpp>
#include <ros/ros.h>
#include <geometry_msgs/PoseStamped.h>
#include <geometry_msgs/Twist.h>
#include <geometry_msgs/TwistStamped.h>
#include <sensor_msgs/Joy.h>
#include <std_msgs/Int32.h>
#include <std_msgs/Float32.h>
#include <Eigen/Dense>
#include <yaml-cpp/yaml.h>
```

Include dependency graph for stewart_controller.hpp:



Classes

- class Stewart

  Stewart Controller Class This class takes care of interfacing with Webots control API and implementing kinematics and trajectory plan for the Stewart Platform.

Macros

- #define N_ITERATIONS 3

  Number of iterations for FK.

5.2.1 Macro Definition Documentation

5.2.1.1 N_ITERATIONS

```
#define N_ITERATIONS 3
```

Number of iterations for FK.

Definition at line 19 of file macros.hpp.

Index