Robot-aided TMS

API

Mainwindow Methods:

- void onLoadClicked();
  Loads the calibration matrices X and Y stored in a .txt file named "calibration.txt" stored in the build folder. Then it saves X and Y values in the global variables T_EE_coil and T_bR_cam.

- void onConnectClicked();
  Open the connection with both the panda robot and the camera, starting the data streaming thread.

- void onDisconnectClicked();
  Update the gui to enable the connection button and to disable the stimulation features.

- void onCameraConnected();
  Update the gui to enable the disconnection button and to enable the stimulation features.

- void onCameraDisconnected();
  Update the gui to change the camera connection color from green to grey.

- void onRobotConnected();
  Update the gui to change the robot connection color to yellow.

- void onRobotDisconnected();

- void onRobotStarted();
  The gui changes the robot connection color to green.

- void onRobotStopped();
  The gui changes the robot connection color to red.

- void onRobotFinished();
  The gui changes the robot connection color to yellow.

- void onRobotError();

The gui changes the robot connection color to red.

- void onAcquirePointClicked();
  Acquire the current coil pose with respect to the head frame and set it as target. Note: this function is used with the SofTaxic interface that streams the coil pose with respect to the head.

- void onLoadPointClicked();
  Read the target coil pose from a .txt file stored in the build folder and named “target.txt”.

- void onSofTaxicPointClicked();
  Acquire the target as the stimulation point currently highlighted in the Softaxic interface

- void onSTARTClicked();
  Emit the goTracking signal.

- void onSTOPClicked();
  Stop the robot.

- void onRestClicked();
  Stop the current session and slowly move the robot to the rest configuration.

- void onNewForceValue();
  Display on the gui the current error in positioning the coil and check if the errors are under the set thresholds. If the errors are higher than the thresholds for more than maxErrorTime, the signal goReArrangeRobot is emitted. The stimulation features are enabled only when the errors are lower than the thresholds.

- void onHeadCompensationStarted();
  The robot is on the head and the stimulation features are enabled, after checking that the serial port to send the TMS trigger is open. The number of the stimulation session is increased by 1. The error threshold values are taken from the gui and stored in the global variables.

- void onHeadCompensationStarted_RE();
  The robot is on the head and the stimulation features are enabled, after checking that the serial port to send the TMS trigger is open. The number of the stimulation session is increased by 1. The error threshold values are taken from the gui and stored in the global variables.

- void onPointSelected();
  Set the current target as the point currently selected in the gui.

- void onEmergencyStop();
  This function is called after an EMERGENCY STOP occurred. The robot label and the emergency label on the gui became red.

- void onStartStimulationClicked();
  Send a trigger to the serial device (connected to the TMS) each X seconds. If the coil positioning error is over threshold wait for the next trigger. The interstimuli time X is acquired from the gui, where can be set from the user.

- void onStopStimulationClicked();
  Stop sending trigger to the TMS.

- void on10StimulationClicked();
  Send 10 triggers to the serial device (connected to the TMS) each X seconds. If the coil positioning error is over threshold wait for the next trigger. The interstimuli time X is acquired from the gui, where can be set from the user.

- void onManualStimulationClicked();
  Enable the manual triggering: the user manually send the trigger pressing the "GO" button on the gui.

- void onConnectSerialPortClicked();
  Open the communication with the selected serial port.

- void onDisconnectSerialPortClicked();
  Close the communication with the currently connected serial device.

- void onSerialDeviceSelection();
  Look for the available serial devices and show them on the gui.

- void SendTriggerTMS();
  Send the trigger to the serial device and update on the gui the number of delivered stimuli for the current session.

- void onPrintData(QVector<double> data);

Save useful robot and error data on file “data.txt”. The file is saved in a new folder with the name specified in the gui in the “Folder name” label.

- void UpdateCameraData(QByteArray data);
  Receive a QByteArray with the Camera data. Extract from the data the position and orientation features for the head and coil item and save them in the global variables. If the head tracking is active compute the error between the actual and the target coil pose.

- void on_pushButtonSavePoint_clicked();
  Save on file (“target.txt”) the currently highlighted target point.

PandaClass Methods:

- void run();

- void initialize();
  Open the connection with the robot and emit the connected() signal if the robot is available.

- void goTracking();
  Robot control loop with a torque control, setting the robot impedance. The target pose is computed as minimum jerk trajectory to move the coil on the target while compensating the head movements. If the head (reference) moves too fast the robot stops and the emergency control is activated. The trajectory is divided in two paths: the first one moves the coil 10 cm on the target orienting the coil with the target orientation, then the second path moves the coil slowly down touching the head.

- void goResting();
  Robot control loop with a cartesian velocities control. The robot moves in a predefined rest pose. If the coil is on the head, the robot first moves slowly the coil 10 cm above the scalp (updating the trajectory with respect to eventual head’s movements) and then moves in the rest pose.

- void goHeadCompensation();
  Robot control loop with a torque control, setting the robot impedance. The target pose is updated with respect to eventual head’s movements each 1 second. Between two updates the robot trajectory is computed as minimum jerk trajectory. If the head (reference) moves too fast the robot stops and the emergency control is activated.

- void onEmergencySTOP();
  Stop the robot and updates the global variables that say if the robot is moving towards the head (“TRACKING” variable) or is on the target point (“onHEAD” variable).

- void onReArrangeRobot();
  Robot control loop with a torque control, setting the robot impedance. The robot moves the coil at a defined distance “DeltaArrangement” above the scalp, in a time equal to “time_arrangement” and then moves the coil again on the scalp on the target pose. The trajectories are computed as minimum jerk trajectories. The target pose is updated with respect to eventual head’s movements.

AUTHOR: Alessia Noccaro, PhD student
Research Unit of Neurophysiology and Neuroengineering of Human-Technology Interaction
Università Campus Bio-Medico di Roma
Via Alvaro del Portillo, 21
00128 Roma, Italy
email: a.noccaro@unicampus.it