Elective in Robotics

Geomagic Touch

Prof. Alessandro De Luca, Dr. Marco Ferro

Dipartimento di Ingegneria Informatica
Automatica e Gestionale Antonio Ruberti

Sapienza
Università di Roma

Geomagic Touch haptic device

two pen
buttons

2 devices available at DIAG Robotics Lab

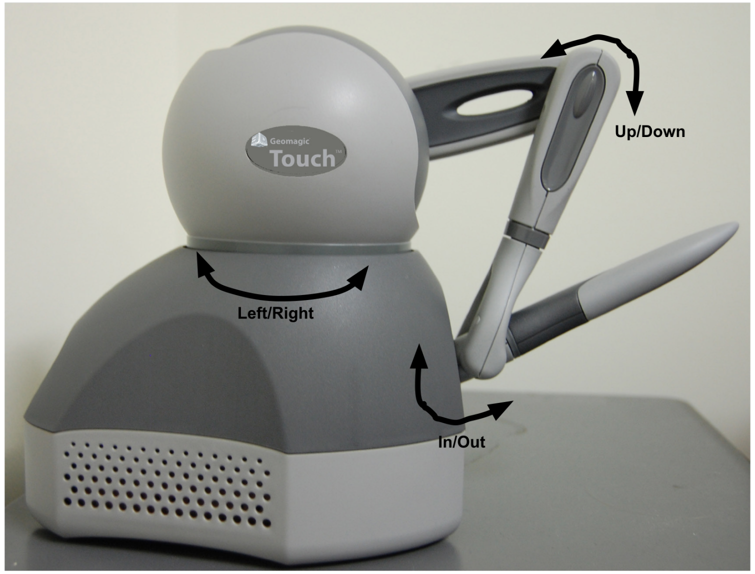
Phantom Omni (same device!)

PHANTOM Omni ⇒ now Geomagic Touch

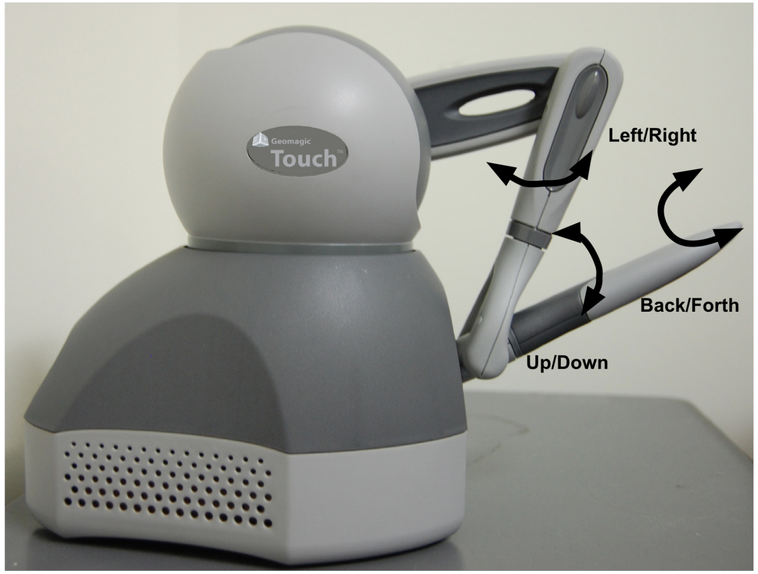
SensAble Technologies ⇒ now 3D Systems

3D SYSTEMS

Range of device motion

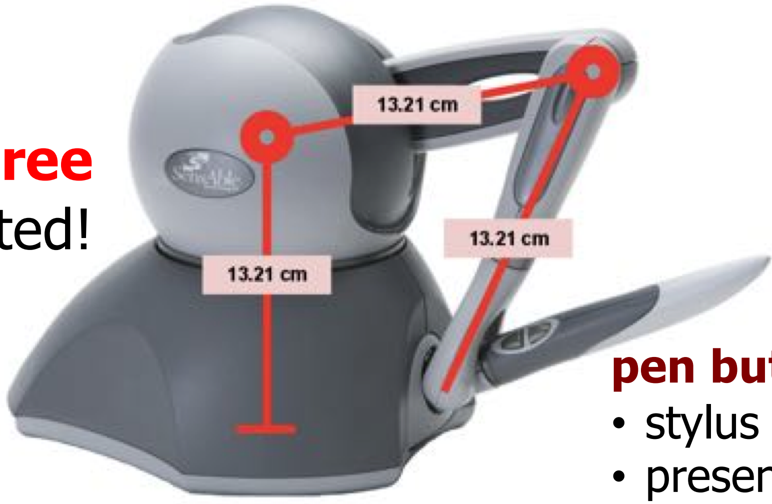


macro movements



micro movements
(of the stylus)
with fixed HIP

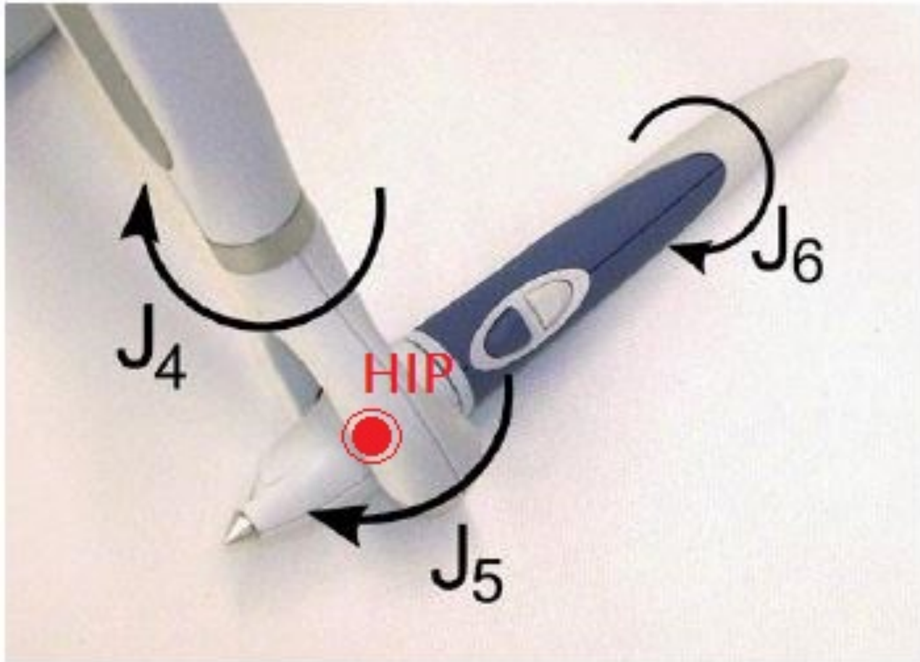
only the **first three** joints are actuated!



pen buttons

- stylus switch ON if pressing **dark** gray button
- presence switch ON if pressing light gray button

Degrees of freedom



first three joints
(positioning of the HIP):
actuated

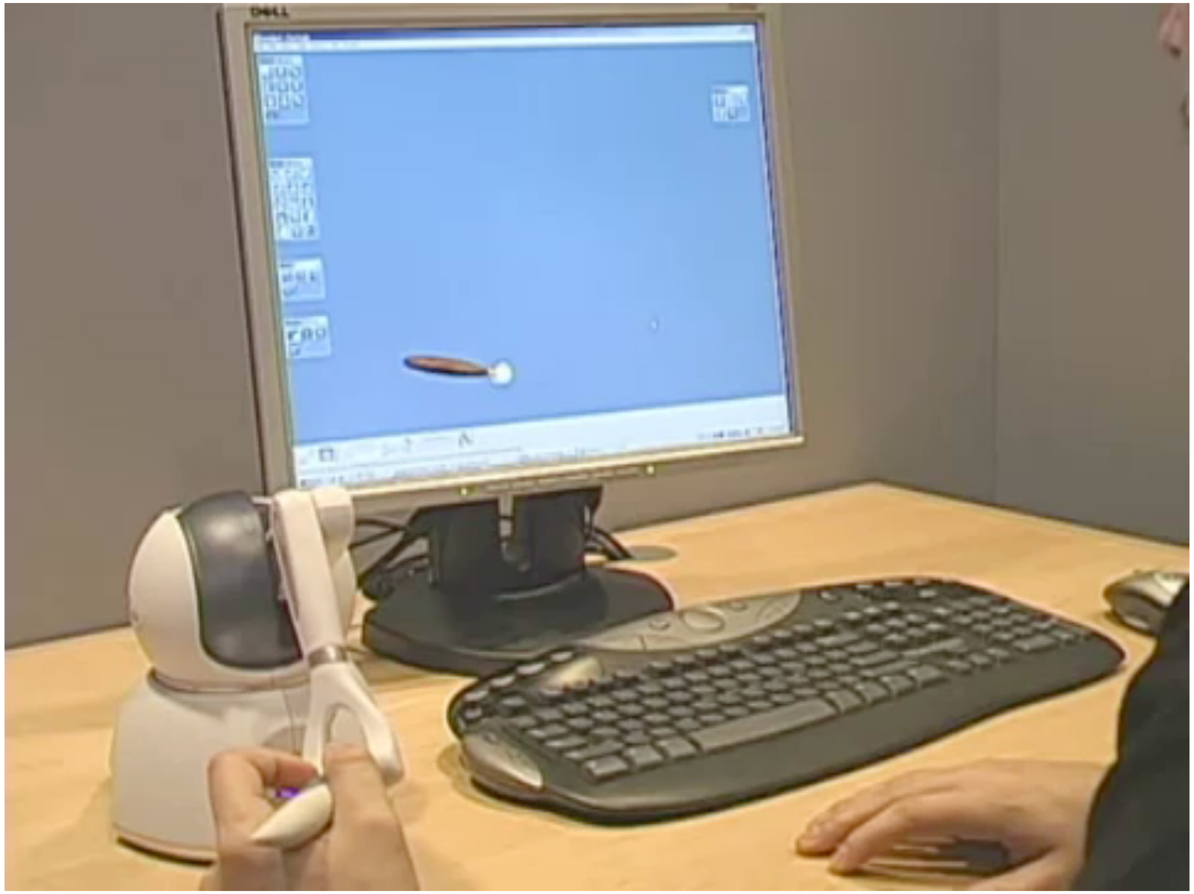
last three joints as spherical wrist
(orientation of the stylus):
passive

Geomagic Touch data sheet

Force feedback workspace	~6.4 W x 4.8 H x 2.8 D in > 160 W x 120 H x 70 D mm
Footprint (Physical area device base occupies on desk)	6 5/8 W x 8 D in ~168 W x 203 D mm
Weight (device only)	3 lbs 15 oz **(1.42 kg)**
Range of motion	Hand movement pivoting at wrist
Nominal position resolution	> 450 dpi ~ 0.055 mm
Backdrive friction	< 1 oz (0.26 N)
Maximum exertable force at nominal (orthogonal arms) position	0.75 lbf (3.3 N)
Continuous exertable force (24 hrs)	0.2 lbf (0.88 N)
Stiffness	X axis > 7.3 lbs / in (1.26 N / mm)
	Y axis > 13.4 lbs / in (2.31 N / mm)
	Z axis > 5.9 lbs / in (1.02 N / mm)
Inertia (apparent mass at tip)	~0.101 lbm (45 g)
Force feedback (**3** Degrees of Freedom)	x, y, z
Position sensing [Stylus gimbal]	x, y, z (digital encoders)
(**6** Degrees of Freedom)	[Pitch, roll, yaw (± 5% linearity potentiometers)
Interface	RJ45 compliant on-board Ethernet Port or USB Port
Supported platforms	Intel or AMD-based PCs
OpenHaptics®Toolkit compatibility	Yes

Geomagic Touch in action

(actually, a Sensable Omni...)



video

https://youtu.be/REA97hRX0WQ

Geomagic Touch in action

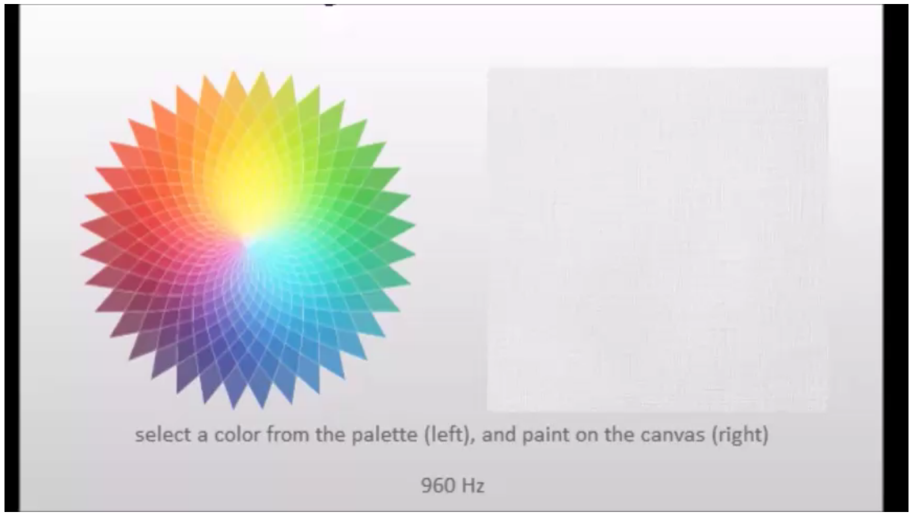
end users

- artists
- industrial designers
- medical professionals
- researchers

applications

- 3D shape modelling/painting
- virtual surgical planning
- virtual navigation
- robot teleoperation
- coordination of robot teams

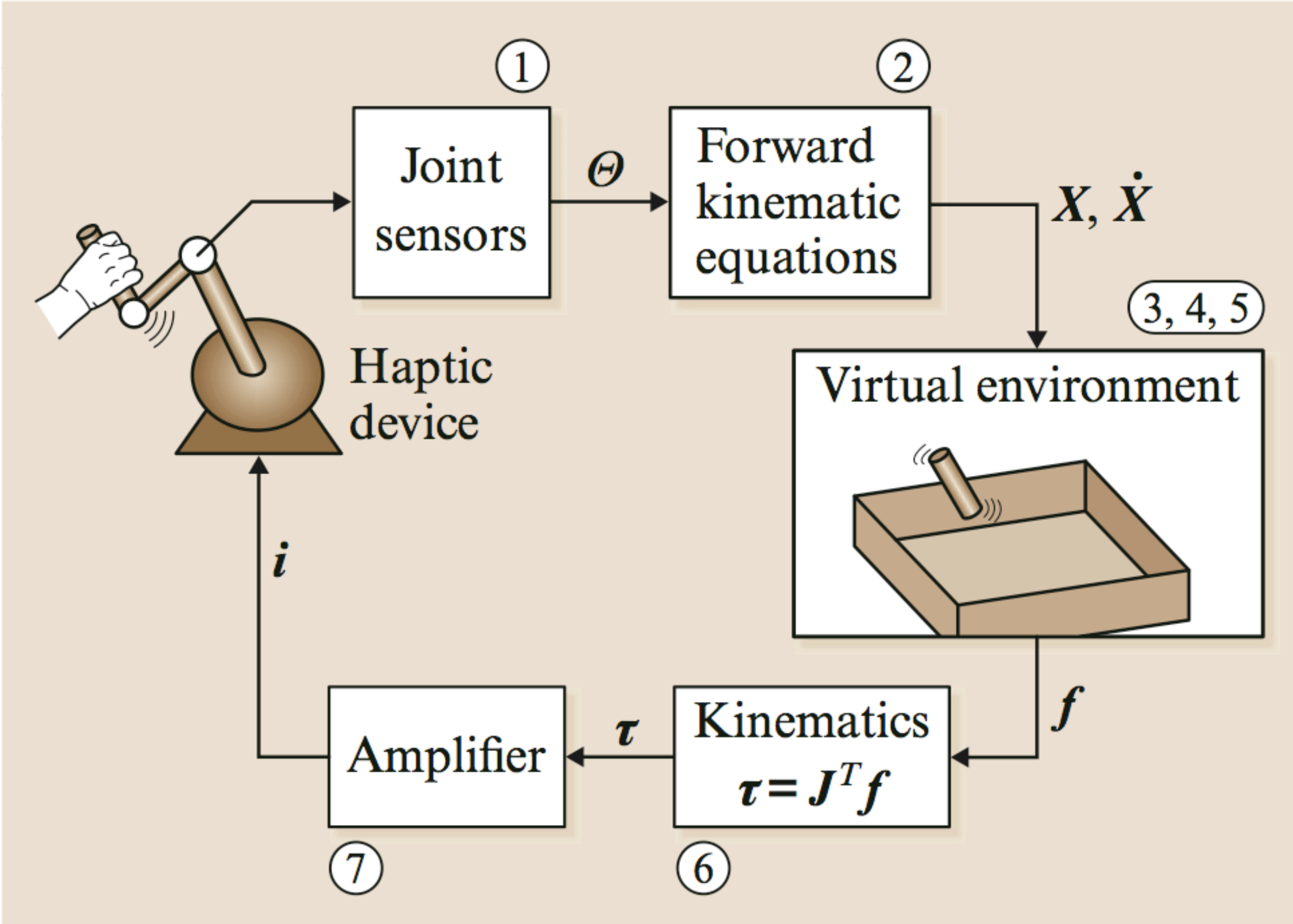
video



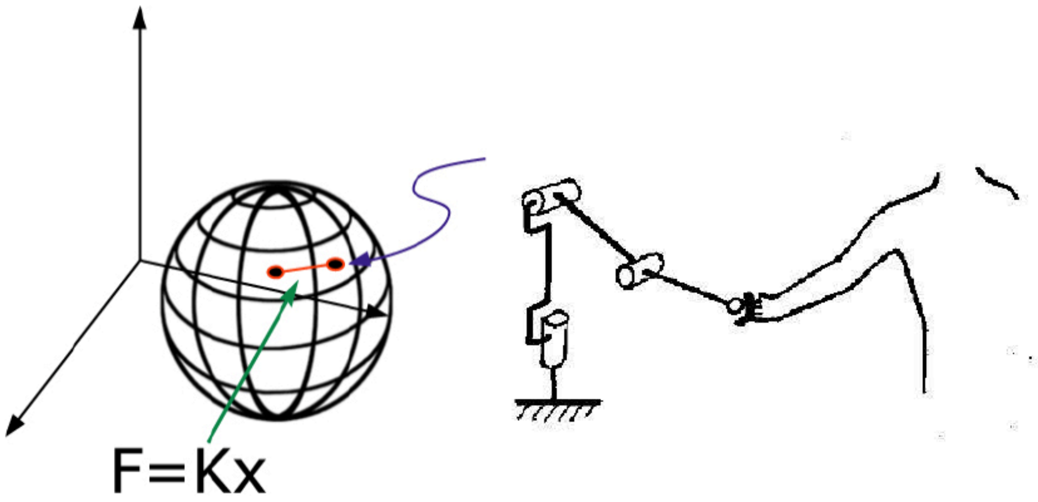
video

Haptic rendering control loop

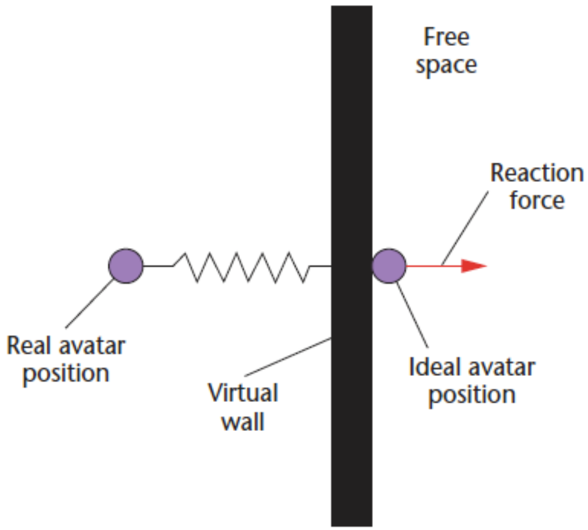
① joint displacement sensing (on device)
② (direct) kinematics
③ collision detection (environment geometry)
④ surface point determination
⑤ force calculation
⑥ kineto-statics
⑦ actuation (on device)

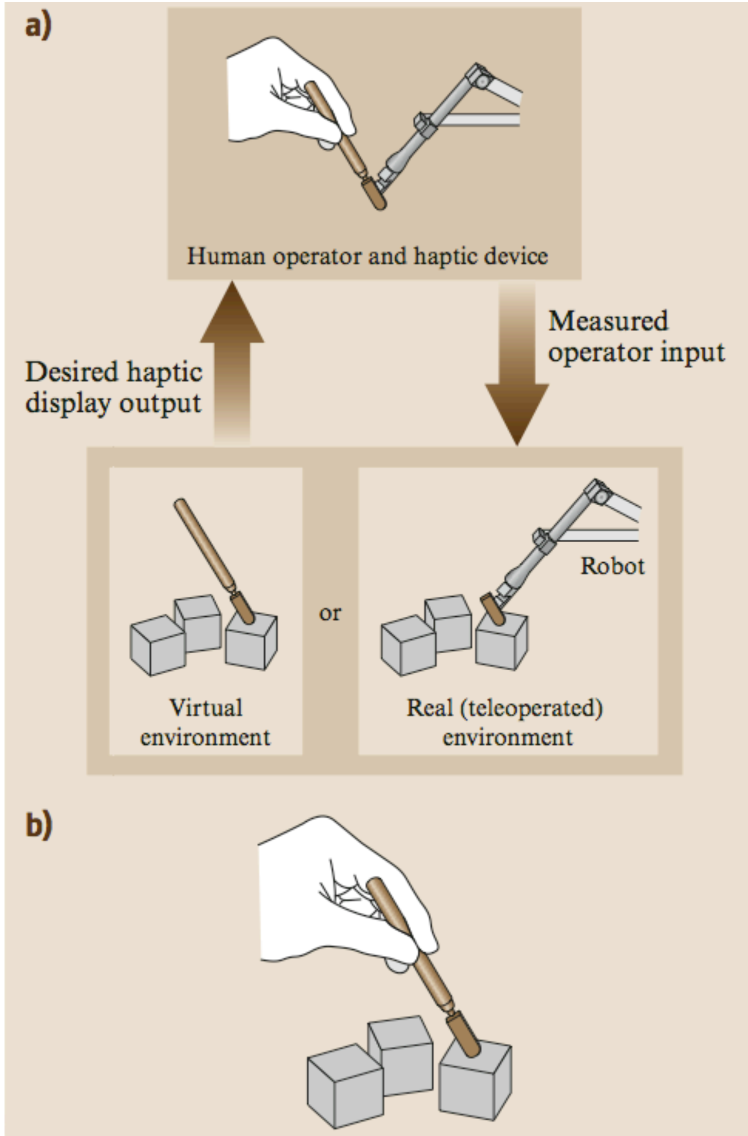


Force feedback from Virtual or Real world



virtual
environment
compliance
modeled with
a spring/damper





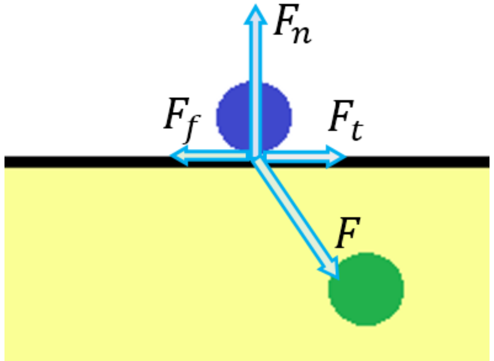
Force feedback from Virtual or Real world

friction forces

- static and dynamic friction forces

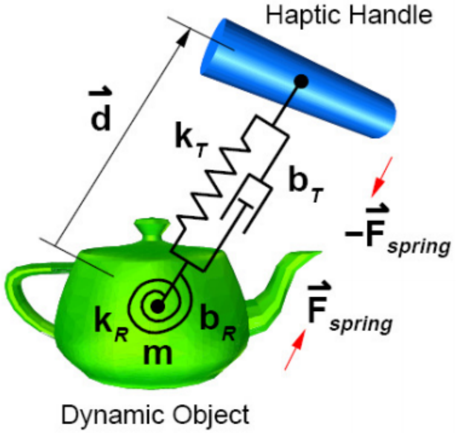
$$F_{f_s} = \mu_s * F_n$$
$$F_{f_d} = \mu_d * F_n$$

no motion if $F_{f_s} > F_t$



virtual contact forces

- “feel” the mass/inertia of a body connected to the virtual tool



Force feedback from Virtual or Real world

force effects

- forces that arise from motion in a portion of the virtual space
- not linked to any virtual object
  - viscous friction
  - stick-slip effect
  - vibration

constraints = virtual fixtures

- attractive/repulsive forces that constrain the virtual tool to a specific geometric region (point, line, plane, surface, cone, …) in the virtual environment
  - elastic/magnetic attraction toward a constraint
  - viscous friction in forbidden directions

Force generation signals

desired force (in software)

counts

D/A

volts

amplifiers

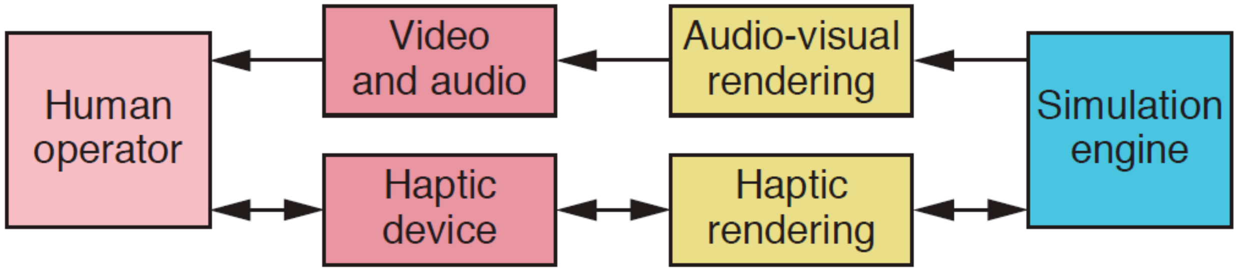
voltage or current

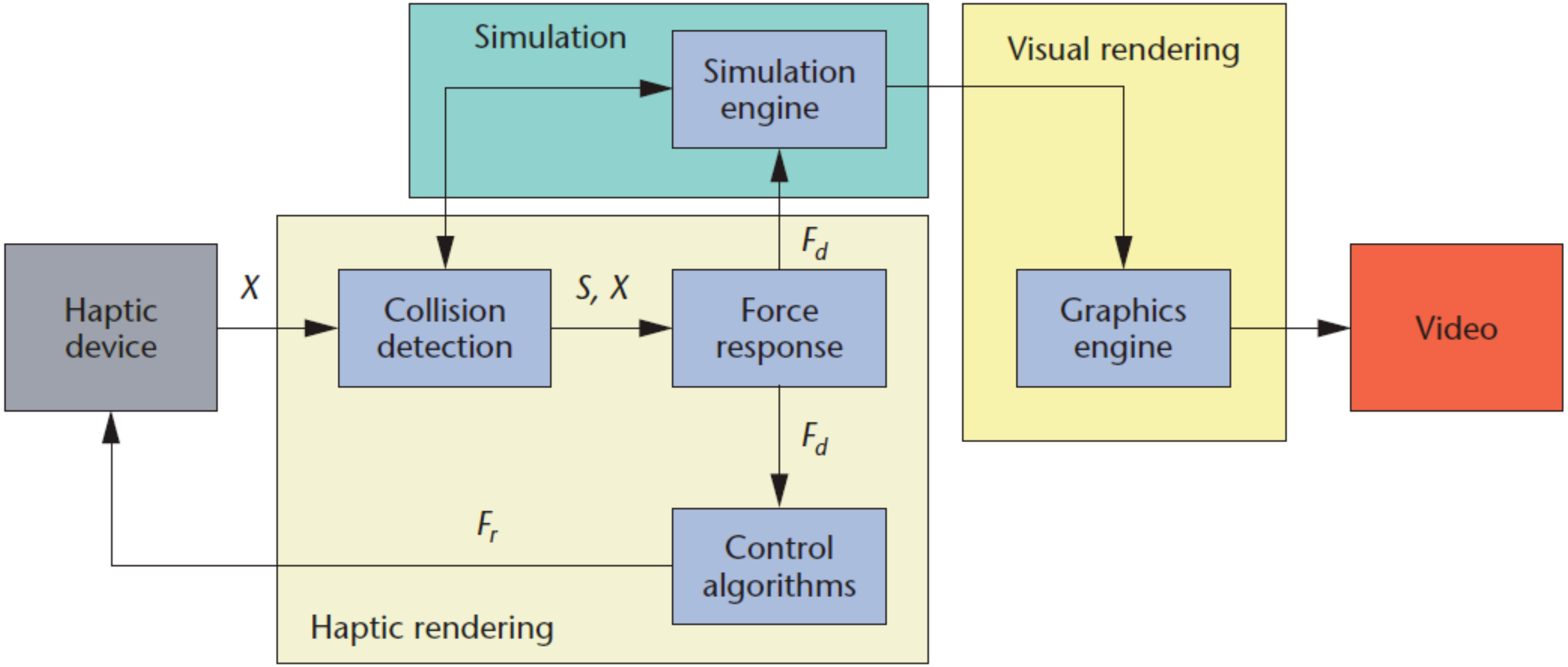
actuator torques

transmissions and kinematics

end-point (3D) force

Haptic/visual rendering architecture



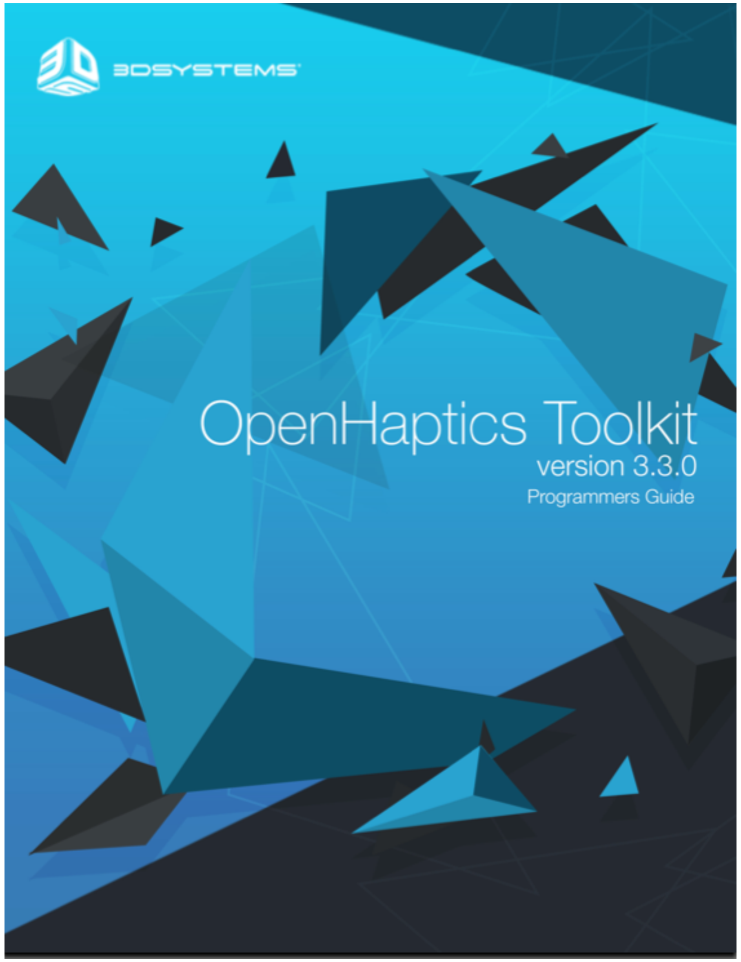


OpenHaptics Toolkit
Library APIs

- proprietary (3DSystems) C++ libraries
  - interfacing with all haptic devices of the Geomagic family
  - development of software for interaction with virtual environments through the device
  - integrated functions for graphic and haptic rendering of a virtual scene
- 3 groups of APIs: **QuickHaptics**, **HLAPI**, **HDAPI**

OpenHaptics Toolkit

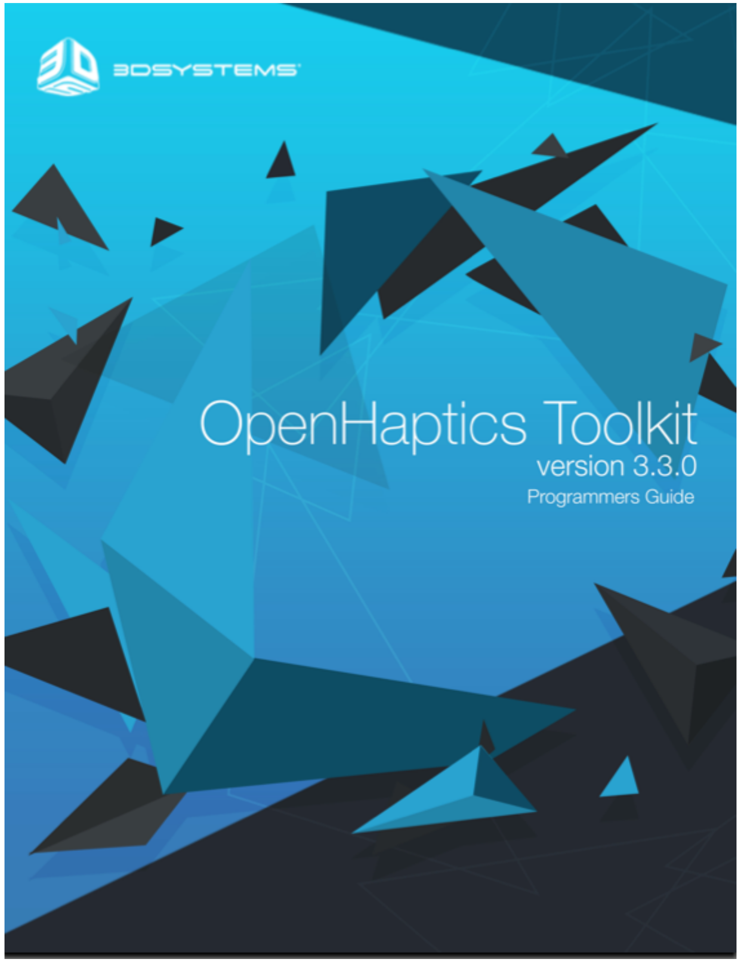
TABLE OF CONTENTS



QuickHaptics API

OpenHaptics Toolkit

TABLE OF CONTENTS



Haptic Device API
=HDAPI

Haptic Library API
=HLAPI

OpenHaptics Toolkit
QuickHaptics micro API

makes programming simpler by encapsulating the basic common steps to all haptic & graphic applications in C++ classes of the QuickHaptics micro API

QuickHaptics properties

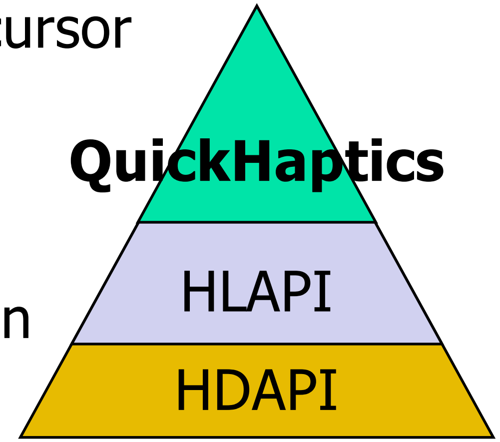
- create simple shapes
- list pre-defined force effects
- define callbacks for contact events

User responsibilities

- create a graphic rendering window
- define device space, shapes with properties, cursor

QuickHaptics responsibilities

- communication with the device
- graphic & haptic rendering and synchronization
- collisions



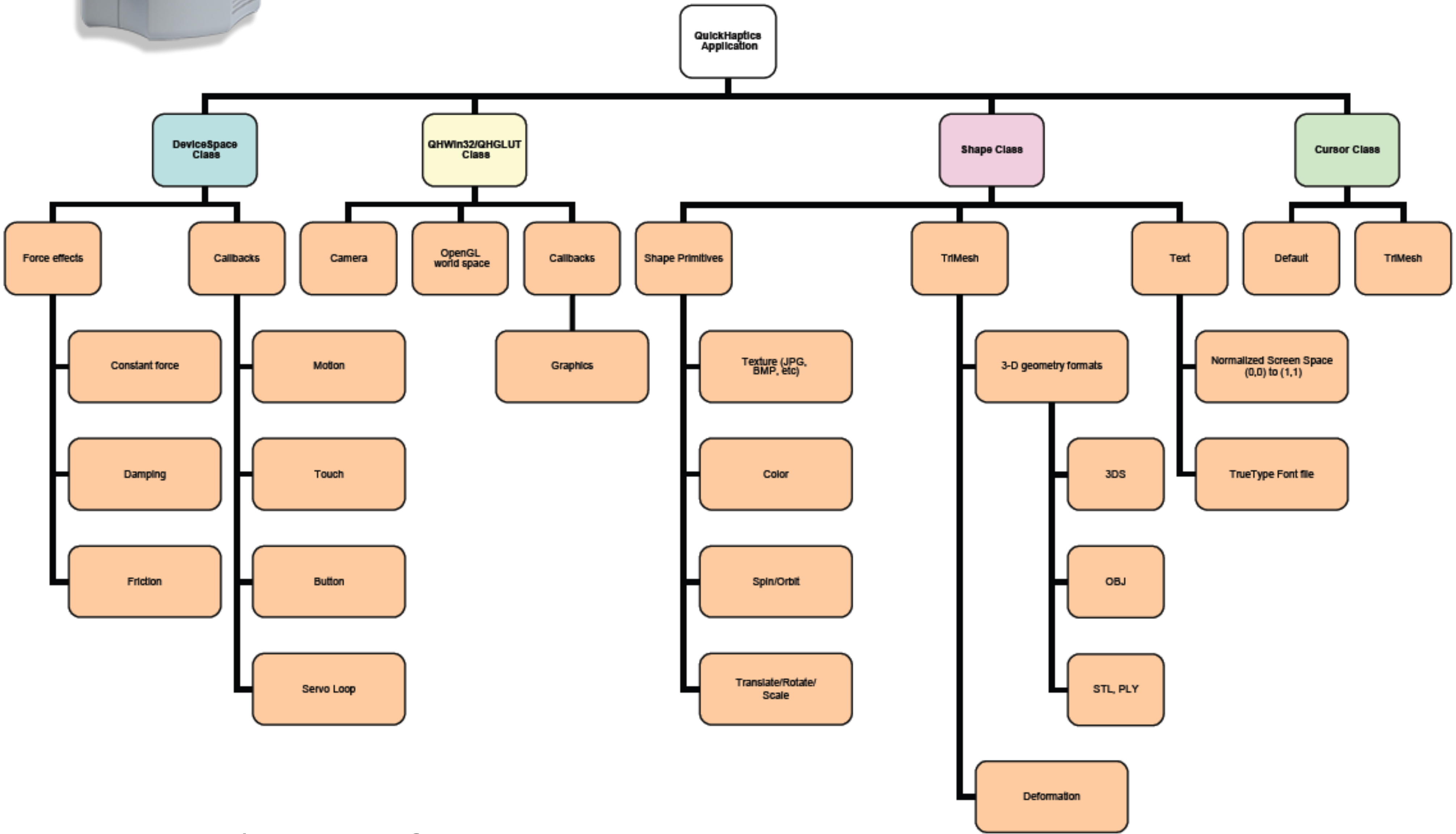
QuickHaptics micro API
classes and properties

implemented in C++, with
4 primary functional classes

location of the
Haptic Interface Point (HIP)
on the Touch device

- DeviceSpace
  - workspace of motion for haptic device
- QHRenderer (OpenGL)
  - on-screen window that renders shapes from a camera viewpoint and lets the user feel those shapes via the device
- Shape
  - base class for one or more geometric objects that can be rendered both graphically and haptically
- Cursor
  - graphical representation of the end point of the second link of the device (HIP)

QuickHaptics micro API
classes and properties



OpenHaptics Toolkit
HLAPI

HLAPI properties

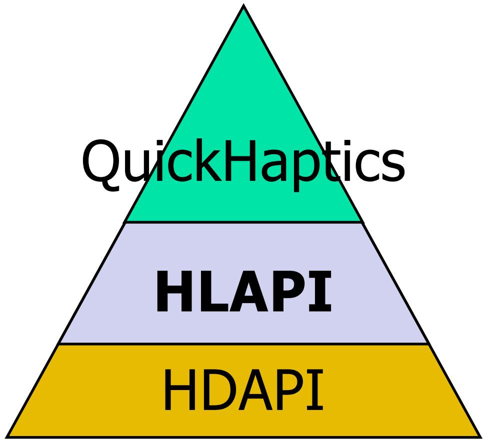
- create graphics and haptic scenes with OpenGL formalism
- define callbacks for arbitrary events
- define constraints (virtual fixtures)

User responsibilities

- define shapes in OpenGL formalism (vertices, edges, transformations, material properties, …)

HLAPI responsibilities

- collision handling and haptic rendering
- communication with the device
- synchronization of graphic and haptic rendering



OpenHaptics Toolkit
HDAPI

HDAPI properties

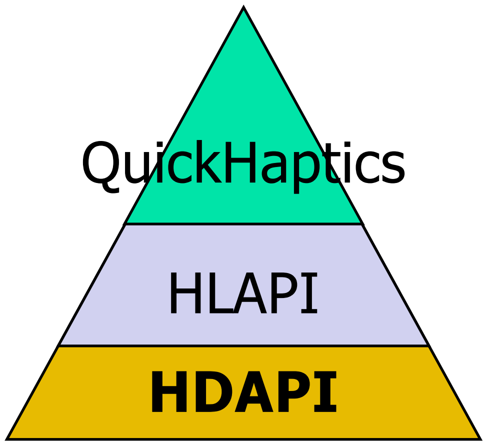
- set user-defined forces to the device
- low-level sensor data readings (joint position and torques)
- define custom haptic rendering, collision detection algorithms

User responsibilities

- graphics rendering, collision detection and haptic rendering
- synchronization management for graphic and haptic loops

HDAPI responsibilities

- initialization and communication with the device



OpenHaptics Toolkit

TABLE OF CONTENTS

Vector Utilities

The <HDU/hduVector.h> header exposes a simple API for common vector operations in three dimensional space.
the functions follows:

Default constructor

hduVector3Dd vec1;

vec1.set(1.0, 1.0, 1.0);

Constructor from three values

hduVector3Dd vec2(2.0, 3.0, 4.0);

Constructor from an array

HDdouble x[3] = {1.0, 2.0, 3.0};

hduVector3Dd xvec = hduVector3Dd(x);

Assignment

hduVector3Dd vec3 = hduVector3Dd(2.0, 3.0, 4.0);

Usual operations:

vec3 = vec2 + 4.0* vec1;

Magnitude:

HDdouble magn = vec3. magnitude();

Dot product:

HDdouble dprod = dotProduct(vec1, vec2);

Cross product:

hduVector3Dd vec4 = crossProduct(vec1, vec2);

Normalize:

vec4.normalize();

Matrix Utilities

The <HDU/hduMatrix.h> header exposes a simple API for common matrix operations.

Default constructor

hduMatrix mat1; // the identity matrix by default

```
HDdouble a[4][4] = {
                    {a1,a2,a3,a4},
                    {a5,a6,a7,a8},
                    {a9,a10,a11,a12},
                    {a13,a14,a15,a16}
                   };
mat1.set(a);
```

Constructor from sixteen values

```
hduMatrix mat(a1,a2,a3,a4,a5,a6,a7,a8,
              a9,a10,a11,a12,a13,a14,a15,a16);
```

Constructor from an array

```
HDdouble a[4][4] = {
                    {a1,a2,a3,a4},
                    {a5,a6,a7,a8},
                    {a9,a10,a11,a12},
                    {a13,a14,a15,a16}
                   };
hduMatrix mat2(a);
```

Assignment

hduMatrix mat3 = mat2;

Get values

double vals[4][4];

mat3.get(rotVals);

Usual operations

mat3 = mat2 + 4.0 * mat1;

Invert

mat3 = mat2.getInverse();

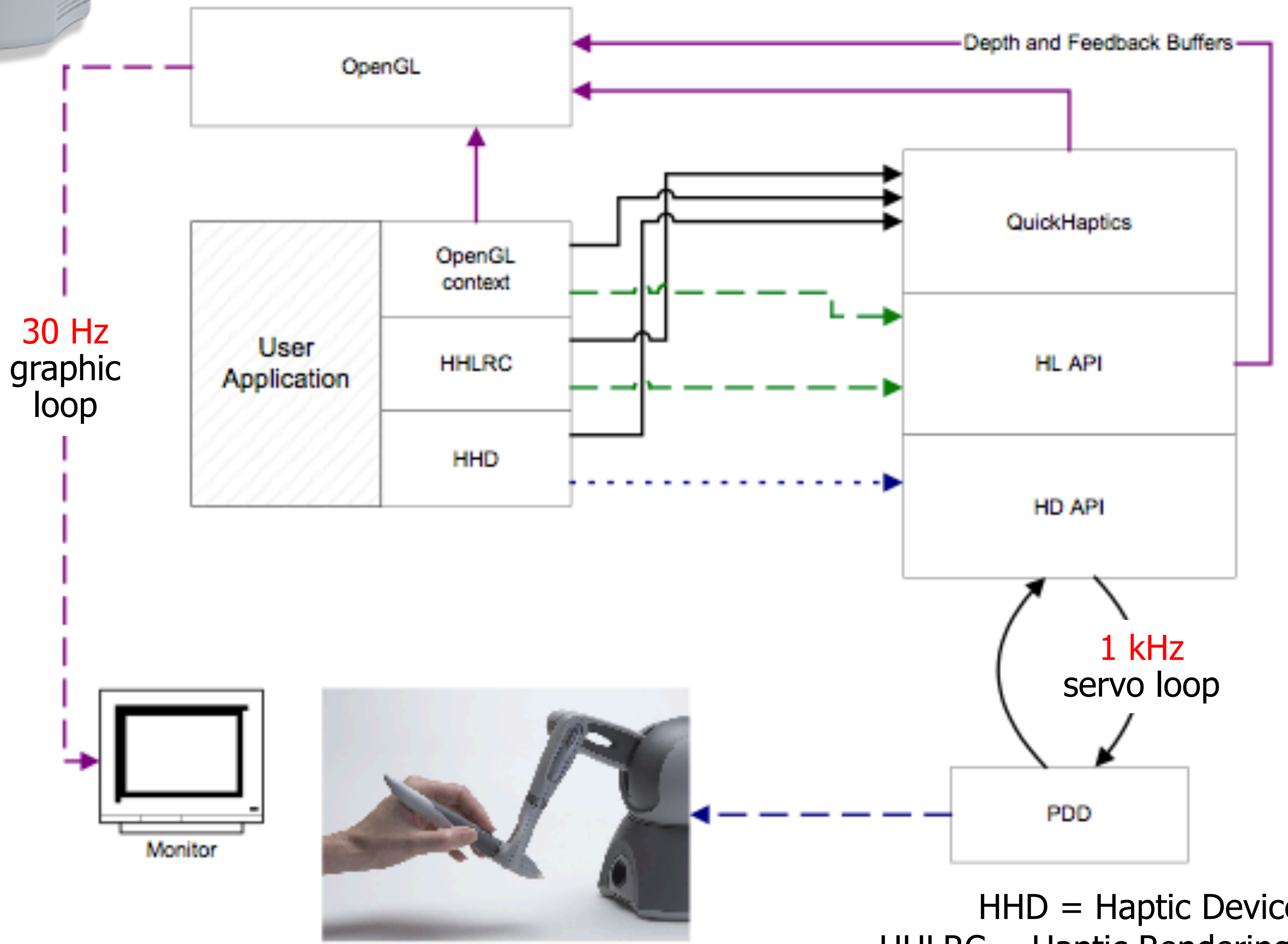
Transpose:

mat3 = mat2.transpose();

Create a rotation

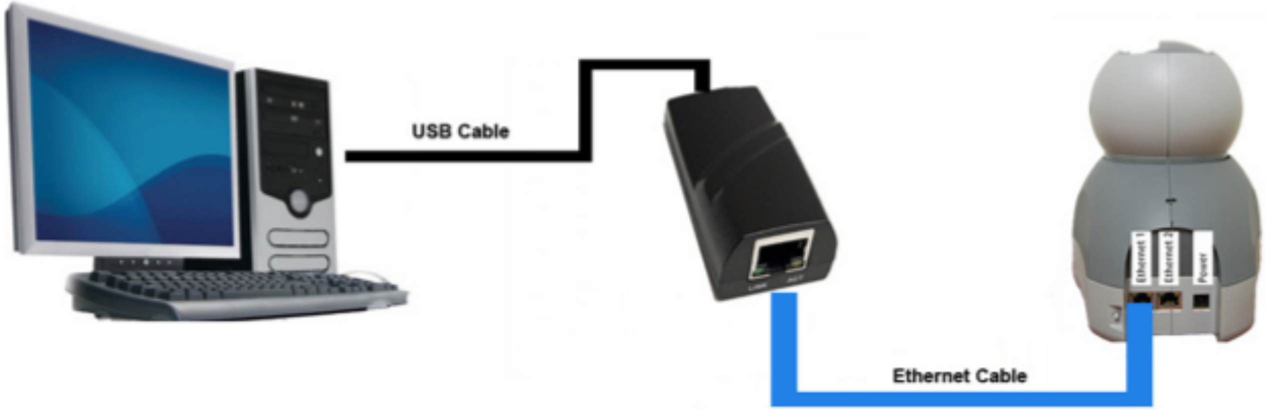
```
hduMatrix rot;
rot = createRotation(vec1, 30.0*DEGTORAD);
HDdouble rotVals[4][4];
rot.get(rotVals);
glMultMatrixd((double*)rotVals);
```

OpenHaptics Overview

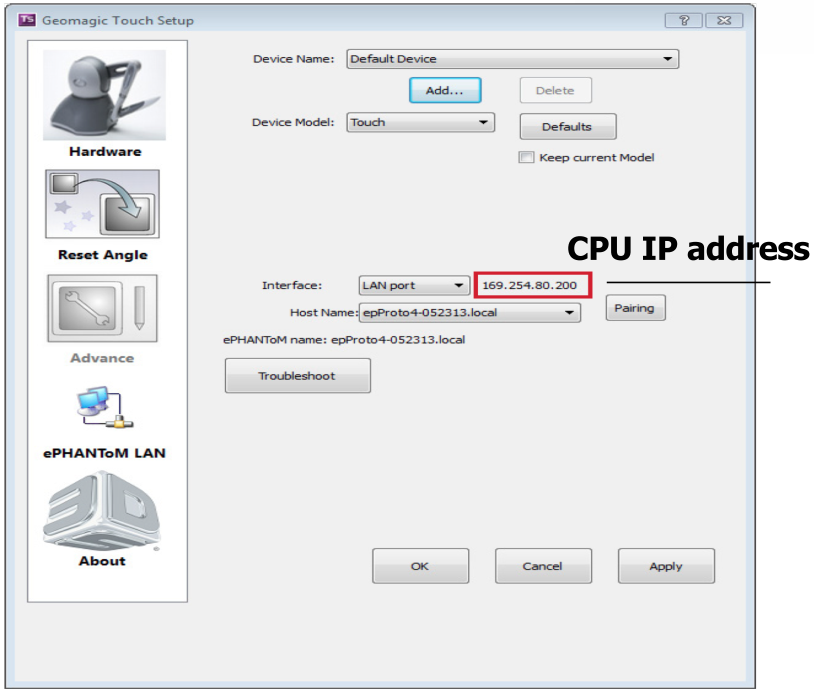


HHD = Haptic Device Handle
HHLRC = Haptic Rendering Context
PDD = "Phantom" Device Drivers = Control

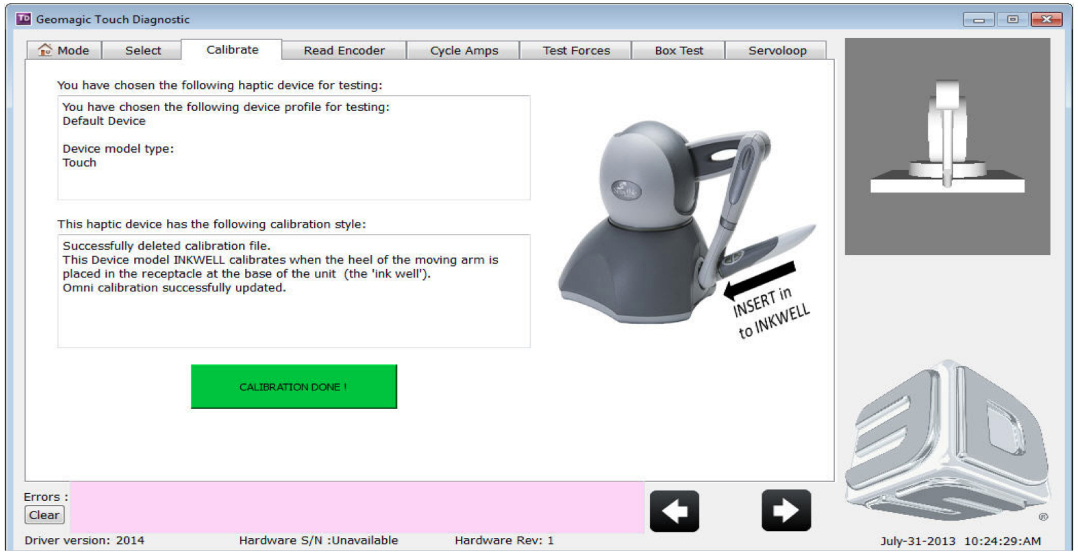
Setting up the system



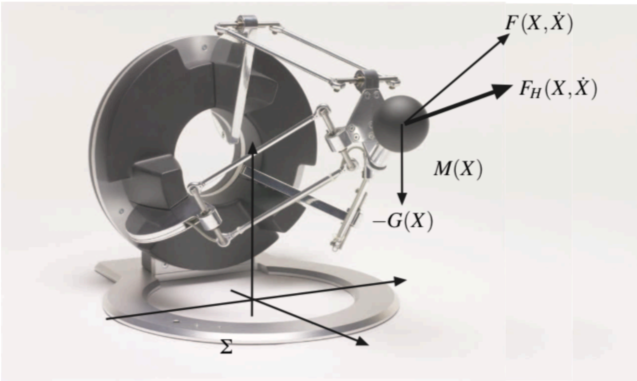
pairing the device



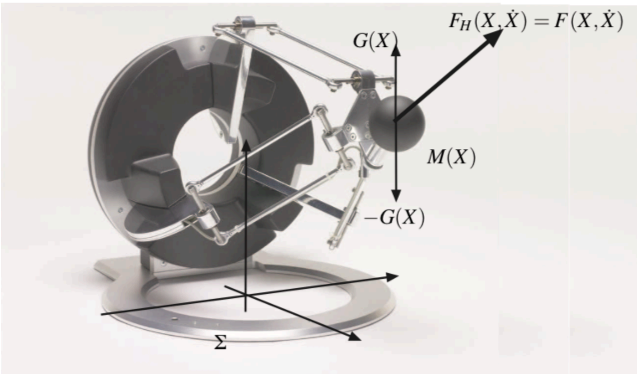
calibrating the device (stylus in the ink well)



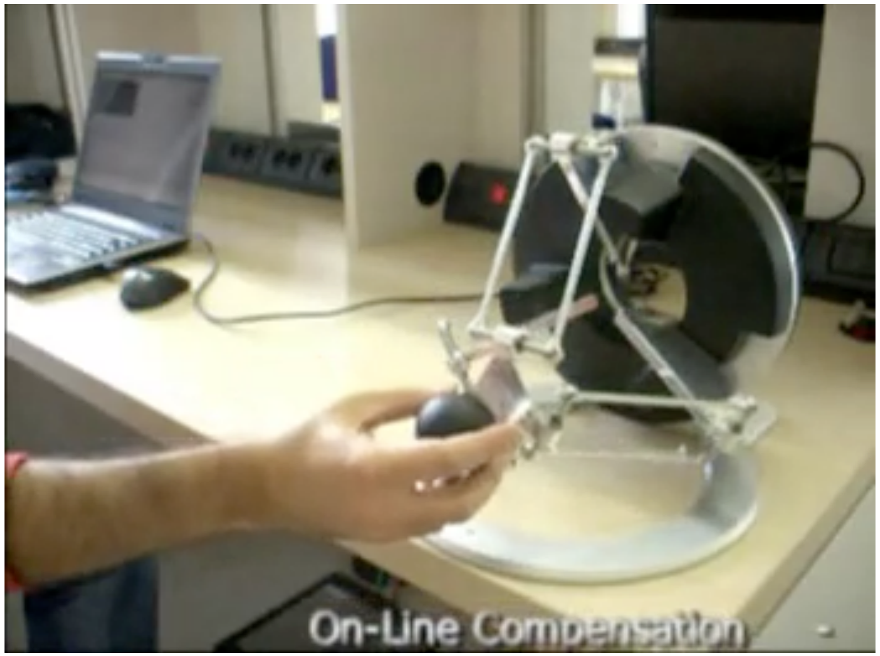
Gravity compensation
self-calibration in many haptic interfaces



- the apparent (position-dependent) gravity acting on end-effector of haptic devices may represent a loss of transparency, requiring its compensation
- off-line self-calibration via an iterative scheme for learning gravity in (many) vertices of a workspace grid, under PD control + on-line compensation



video

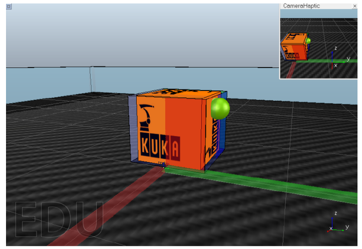


A. Formaglio, S. Mulatto, D. Prattichizzo "Iterative estimation of the end-effector apparent gravity force for 3DoF impedance haptic devices," *10th European Control Conference*, Budapest, 2009, pp. 537-542

based on

A. De Luca, S. Panzieri "An iterative scheme for learning gravity compensation in flexible robot arms," *Automatica*, vol. 30, no. 6, pp. 993–1002, 1994

Geomagic Touch for Robotics
interaction with V-REP/CoppeliaSim



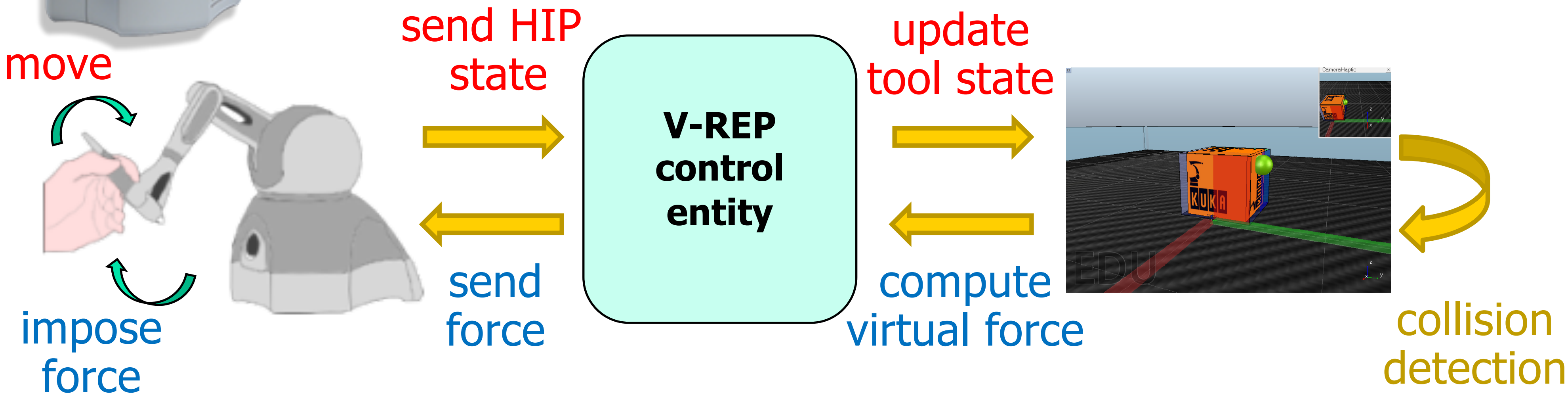
V-REP = Virtual Robot Experimentation Platform

- first release in March 2010; since November 2019 → **CoppeliaSim**

- versatile and flexible simulator for robot programming and control
- multiple programming control entities, with a huge set of dedicated APIs
  - embedded scripts (Lua)
  - **plug-in (C++)**
  - ROS node (C++, Python)
  - **remote API client application (C++, Python, MATLAB)**
  - …

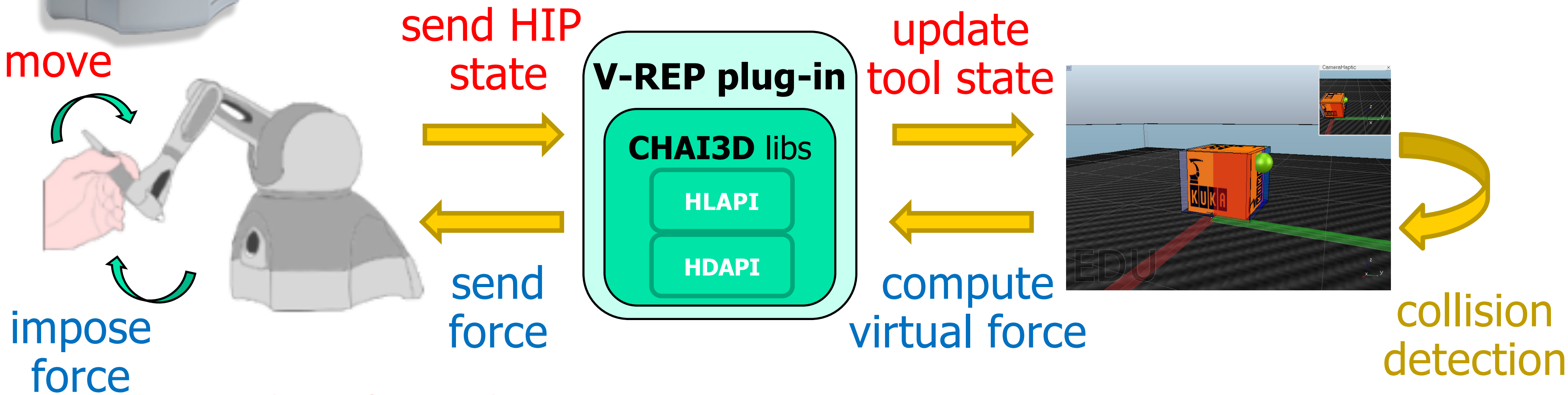
feasible options to interact with **Geomagic** and **OpenHaptics** libraries!

Geomagic Touch for Robotics
interaction with V-REP/CoppeliaSim



- control entity responsible of the communication between the Touch device and the virtual scene
- Haptic Interface Point (HIP) on the device **coupled** with a virtual object in the scene
- multiple choices for feedback force
  - user-defined
  - generated from built-in V-REP **collision detection** module

Geomagic Touch for Robotics
CHAI3D library



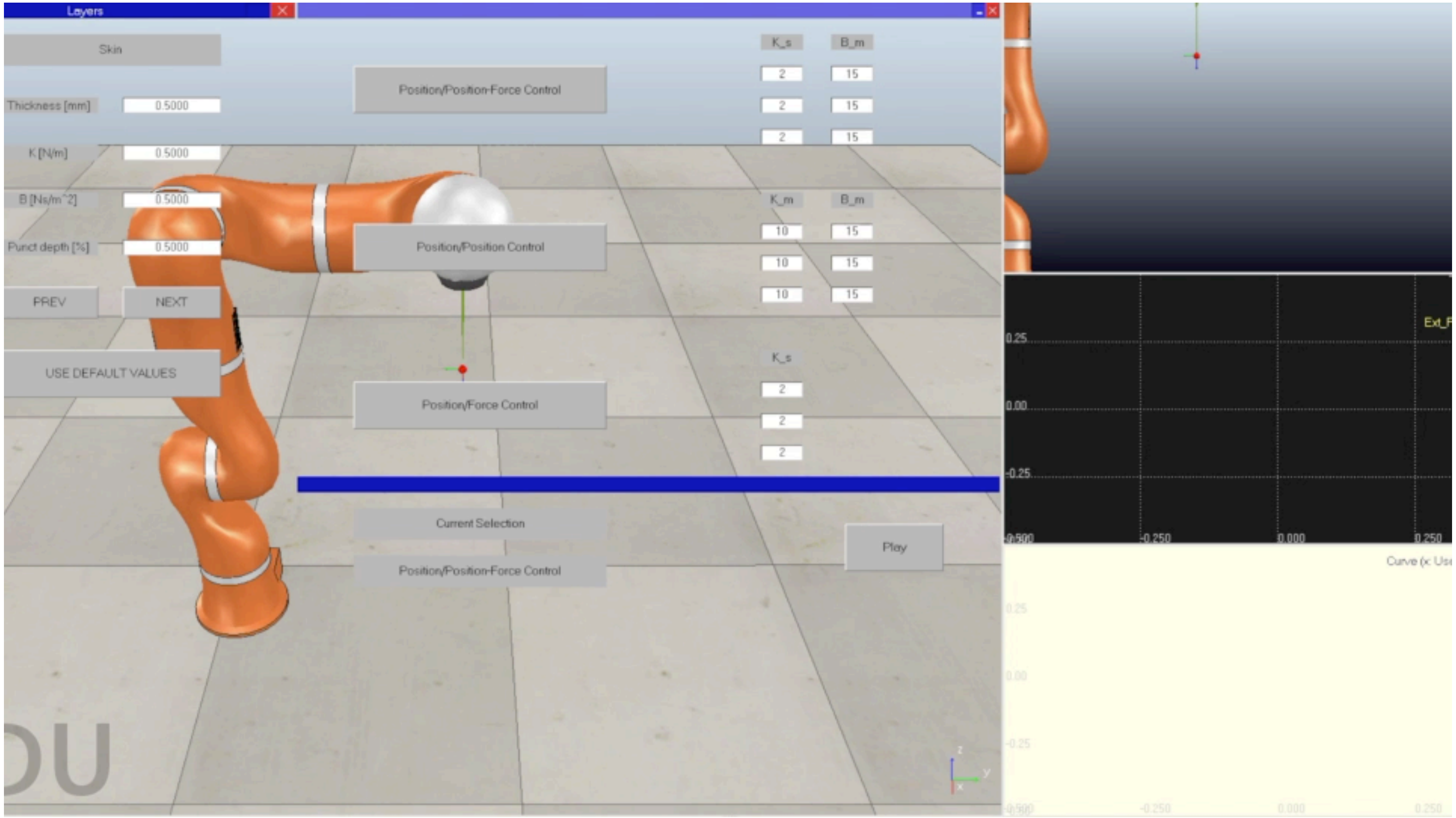
CHAI3D = COMPUTER HAPTICS AND ACTIVE INTERFACE

- multi-platform C++ libraries for haptics and visualization
- wrapper for OpenHaptics libraries
- OpenGL-based graphic and haptic rendering management
- easy get/set of device joints/motors values
- integration module for V-REP as **C++ plug-in**
  - apparently not maintained recently and no more compatible with recent V-REP versions (>3.3.2)!

chai3d

...OBSOLETE!

Needle insertion with force sensing
simulation environment in V-REP



video

- KUKA LWR4, 7-dof robot manipulator
- CHAI3D synchronizes Geomagic HL and VREP simulation loop

Playing with buttons…

- master and slave (e.g., the KUKA LWR4 robot) typically have different workspace sizes
- a clutching mechanism is required to extend Geomagic limited workspace
- this is done using the two buttons on the device in **three** combinations

Button 1

- fully releases the clutch between master and slave
- slave **position** is modified
- slave **orientation** is modified

Button 2

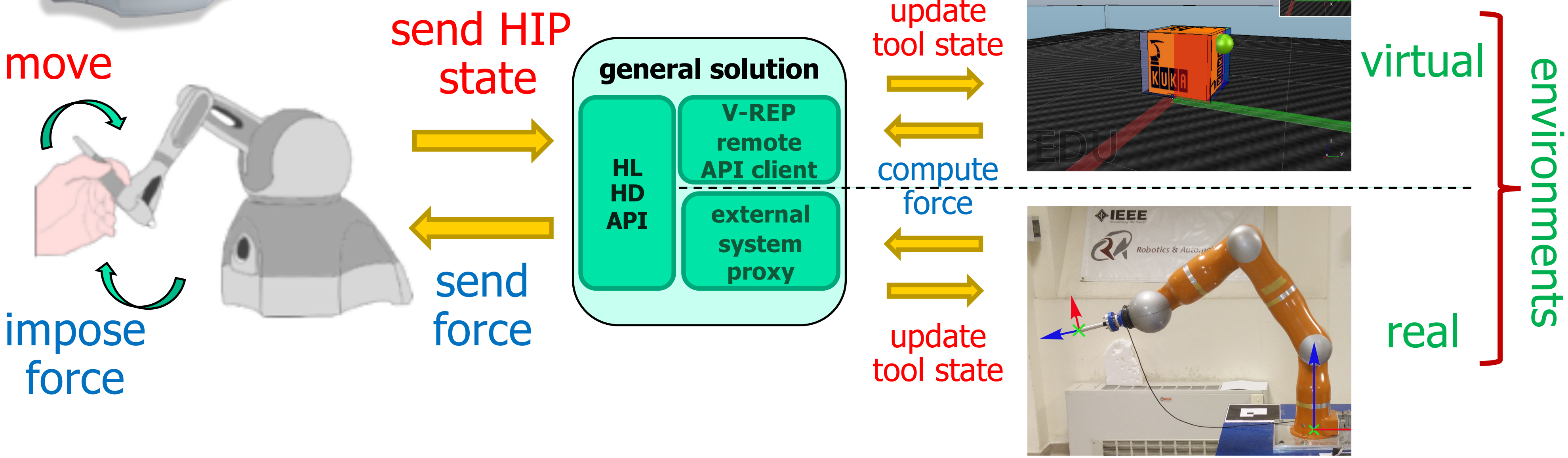
- partially releases the clutch between master and slave
- slave position is not modified
- slave **orientation** is modified

Button 1+2

- simulates a **virtual fixture**
- motion is allowed only along the ***z*-axis** of the slave EE (approach direction)
- slave orientation is not modified

Geomagic Touch for Robotics

a general programming solution

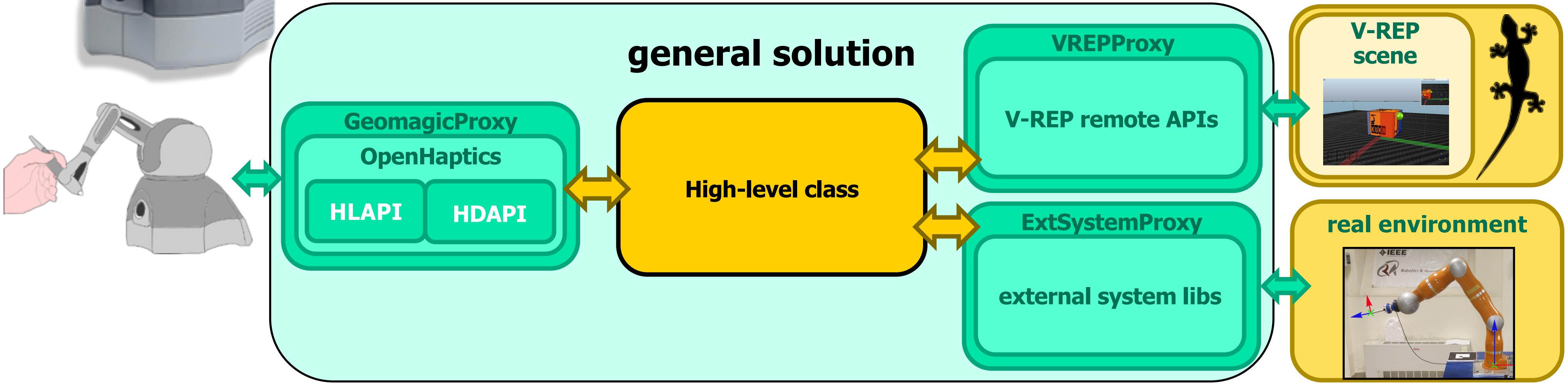


<u>idea</u>: replace CHAI3D with a **general architecture**

- the designed control entity works as a wrapper
  - for OpenHaptics library, to communicate with Geomagic Touch
  - for V-REP remote API client, to interact in virtual environments
  - for arbitrary external systems, to interact in real environments

no need to edit code when moving from simulation to real experiments!

Geomagic Touch for Robotics
a general programming solution



GeomagicProxy class

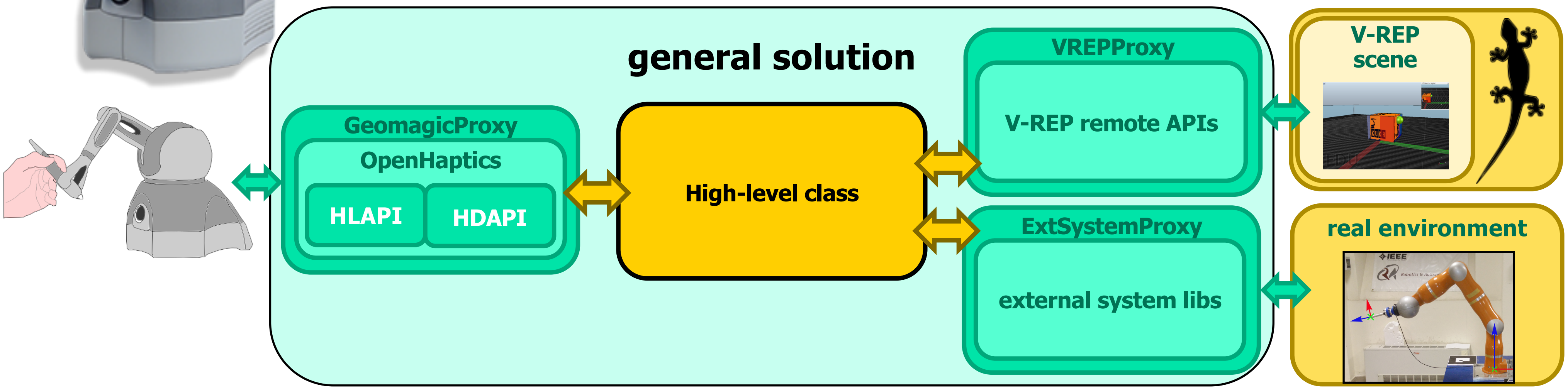
- query OpenHaptics to update the HIP state (pose, 6D velocity)
- request actuation of the haptic device motors to render the input force
- send information to the high-level classes for processing

High-level class

- process and exchange data between GeomagicProxy and VREPProxy class
- set/get values for/from virtual objects in the V-REP scene

Geomagic Touch for Robotics

a general programming solution

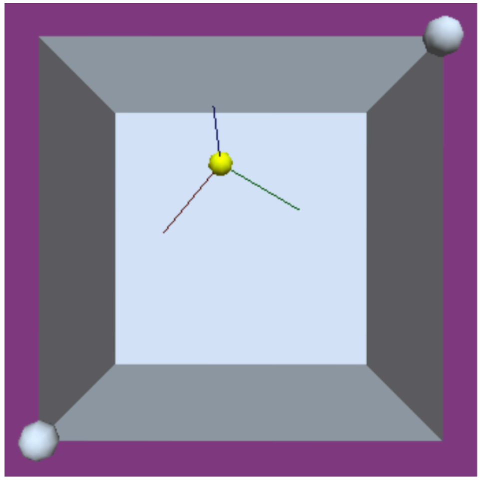



```
/** GeomagicProxy.hpp **/
HDCallbackCode HDCALLBACK updateStateCallback(void* data); // internally set HapticStatus
HDCallbackCode HDCALLBACK forceFeedbackCallback(void* data); // send force commands to the device motors
class GeomagicProxy{
    //...
    struct HapticStatus{
        hduVector3d stylusPosition;
        hduVector3d stylusOrientation;
        hduVector3d stylusLinearVelocity;
        hduVector3d stylusAngularVelocity;
        hduVector3d force;
        int stylusButtons;
    } geoStatus;

    inline HapticStatus getHapticStatus() {return this->geoStatus;}
    inline void setHapticForce(const hduVector3d& f) {this->geoStatus.force = f;}
}
```

```
/**** GeomagicProxy.cpp ****/
// Callbacks call
hdScheduleAsynchronous(forceFeedbackCallback, this, HD_MAX_SCHEDULER_PRIORITY);
while(geo.isRunning()){ // isRunning() evaluates a boolean condition to run the Geomagic thread
    hdScheduleSynchronous(updateStateCallback, this, HD_DEFAULT_SCHEDULER_PRIORITY);
}

/** Any high-level class «using» GeomagicProxy and VREPProxy classes **/
GeomagicProxy geoProxy;
VREPProxy vrep;
while(ctrl.isRunning()){
    // Read the current status of the Geomagic device
    HapticStatus status = geoProxy.getHapticStatus();
    // Set the force vector on the Geomagic device
    geoProxy.setHapticForce(cmdForce); // assume cmdForce computed somewhere else
    // (Example) Transfer the Geomagic stylus motion on a virtual object in V-REP
    vrep.setObjectPosition(objectID,status.stylusPosition);
}
```

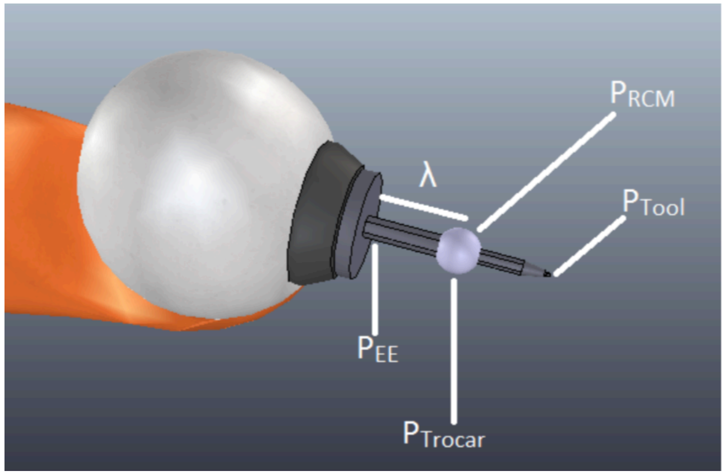
Box constraints and RCM
haptic rendering with Geomagic Touch & KUKA LWR4

definition of a Box around the robot E-E



video

video



definition of a RCM = Remote Center of Motion

What else could be studied?

https://www.quanser.com/products/omni-bundle

- DH parameters, forward/inverse kinematics
- Jacobian matrix and singularities
- joint level PD and PID control
- trajectory planning (joint space vs. task space)
- various haptic (force) rendering laws
  - force fields, "god point", hard and soft contacts

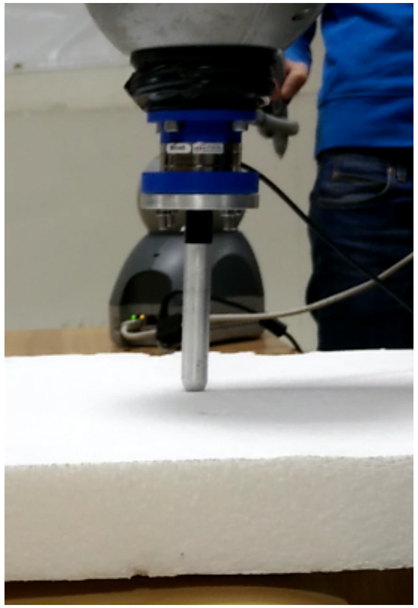
available, e.g.,
in Omni Bundle
by Quanser
with interface to
Simulink/Matlab

- a number of very useful applications…

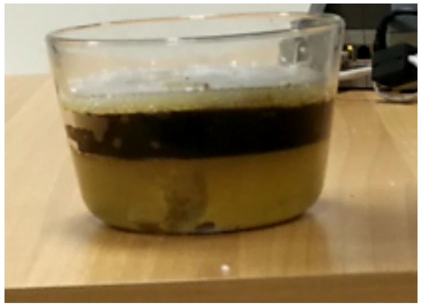
on-going development of a software environment
for the simulation of the haptic device...

Medical applications

Needle insertion with force feedback
teleoperation with Geomagic Touch & KUKA LWR4

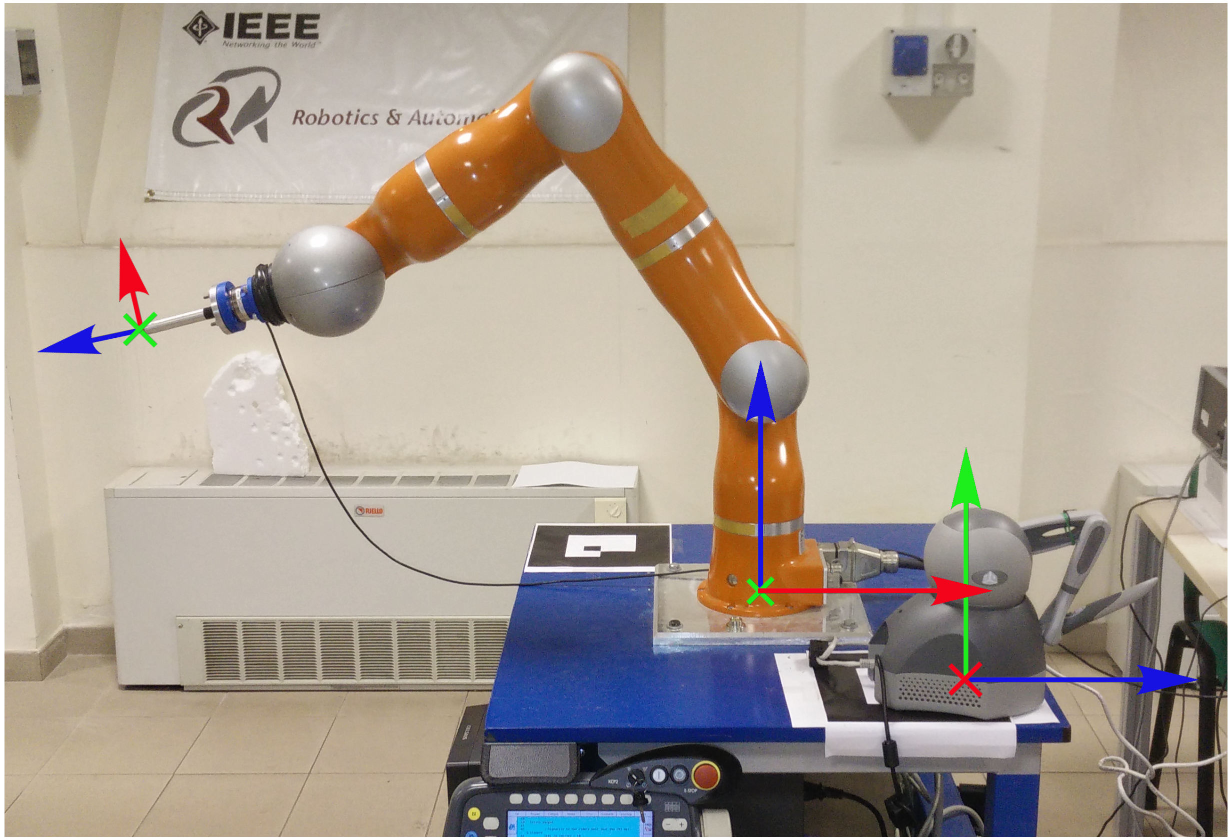


contact with polystyrene



contact with gel layers

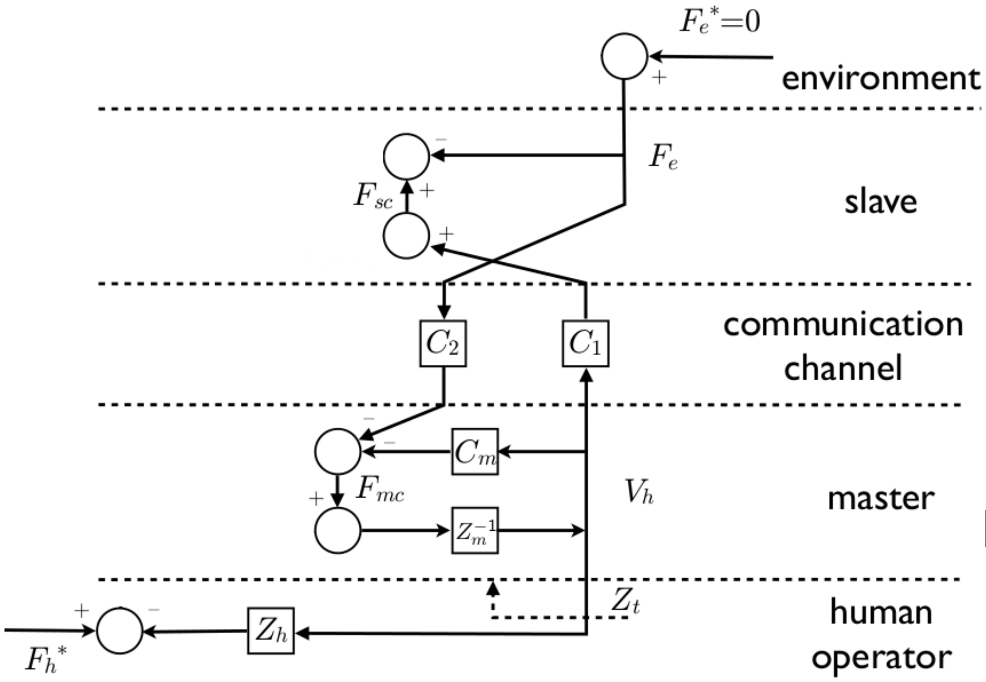
reference frames (z axis = blue)



slave = KUKA
robot manipulator

master = Touch
haptic device

Teleoperation control
2-way and final 3-way implemented solution



basic bidirectional (2-way) scheme

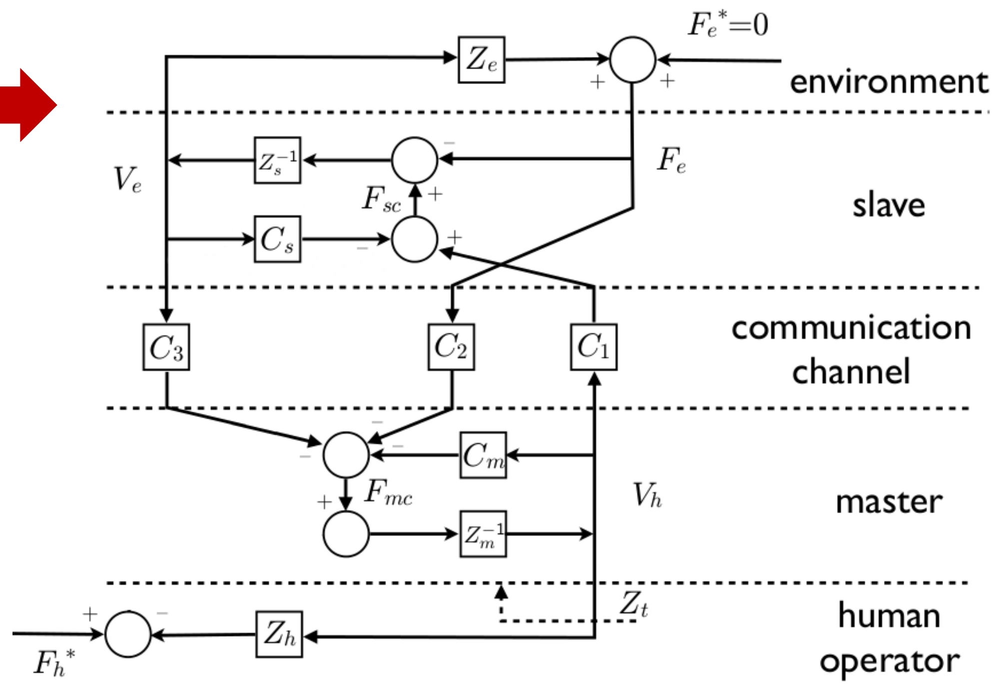
- C_1 = velocity command sent to the slave
- slave control F_{sc} given by the KUKA controller
- C_2 = feedback of sensed force F_s to the master
- design of the master control F_{mc} only

to improve transparency, a 3-way scheme

- additional C_3 = slave velocity feedback
- design of the master control F_{mc} as

$$F_{mc} = K_s F_s + K_v (V_m - V_s)$$

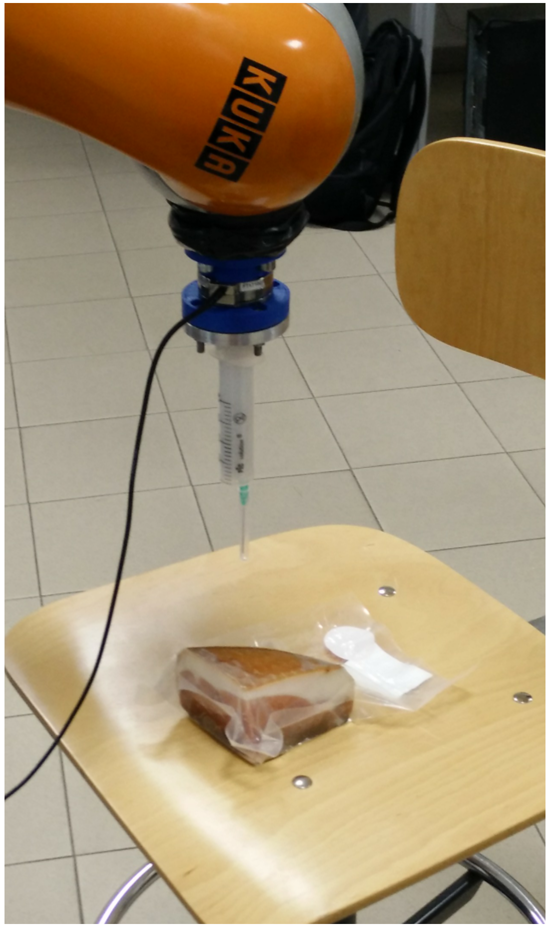
- smoothing (3-sample average) & low-pass filtering of velocity commands @200Hz rate



Needle insertion with force feedback

teleoperation with Geomagic Touch & KUKA LWR4

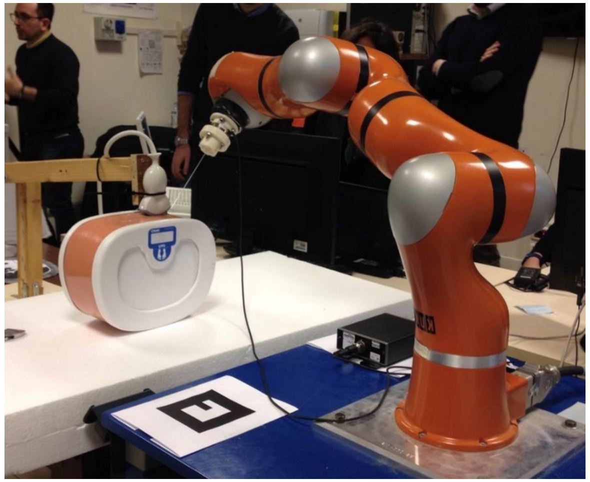
video

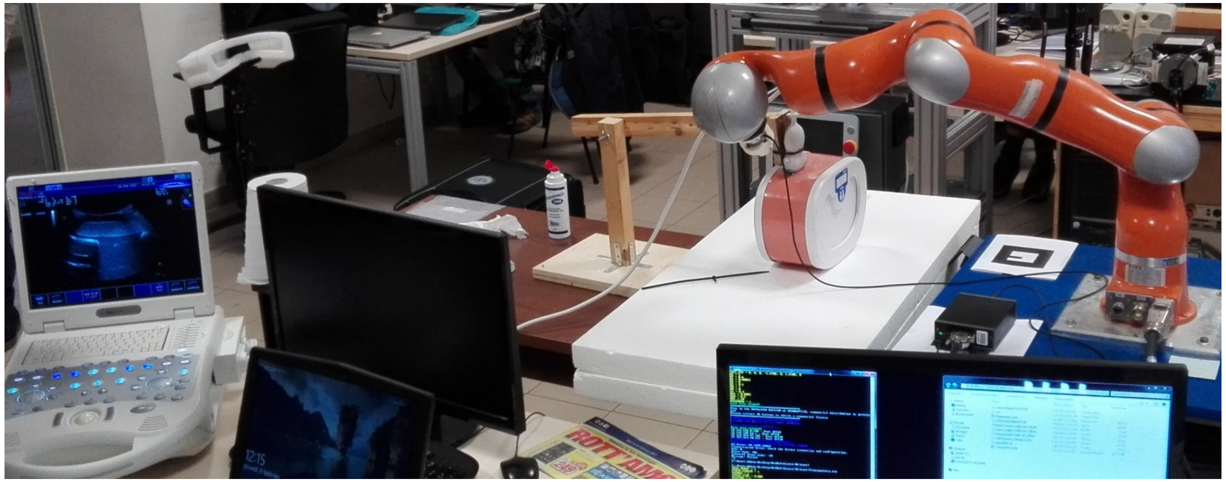


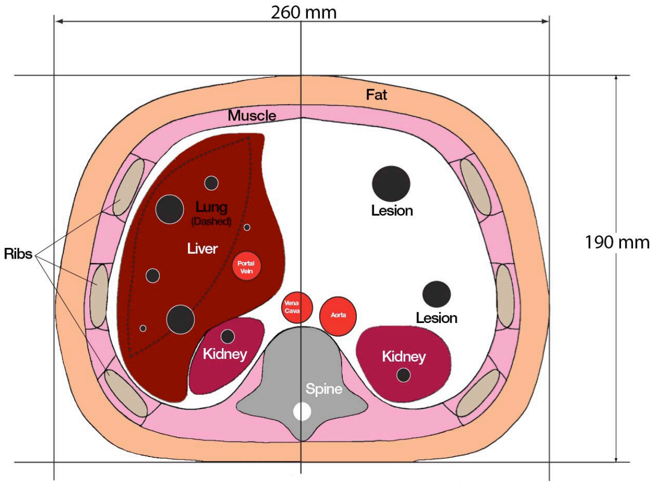
final setup: needle and ham; 3-way teleoperation with force/velocity feedback

Needle insertion steering

teleoperation with Geomagic Touch & KUKA LWR4

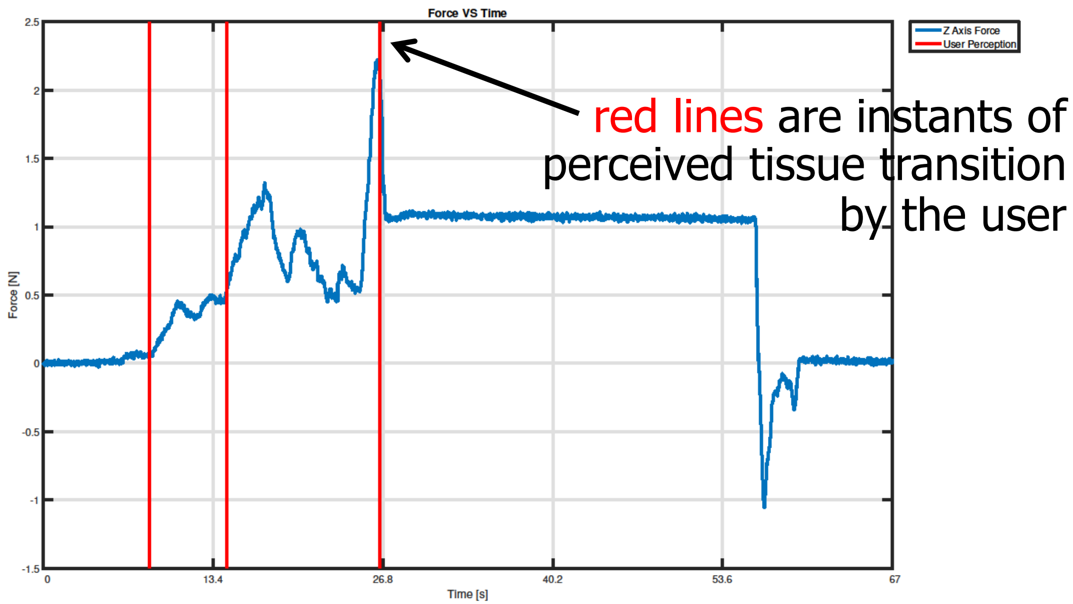




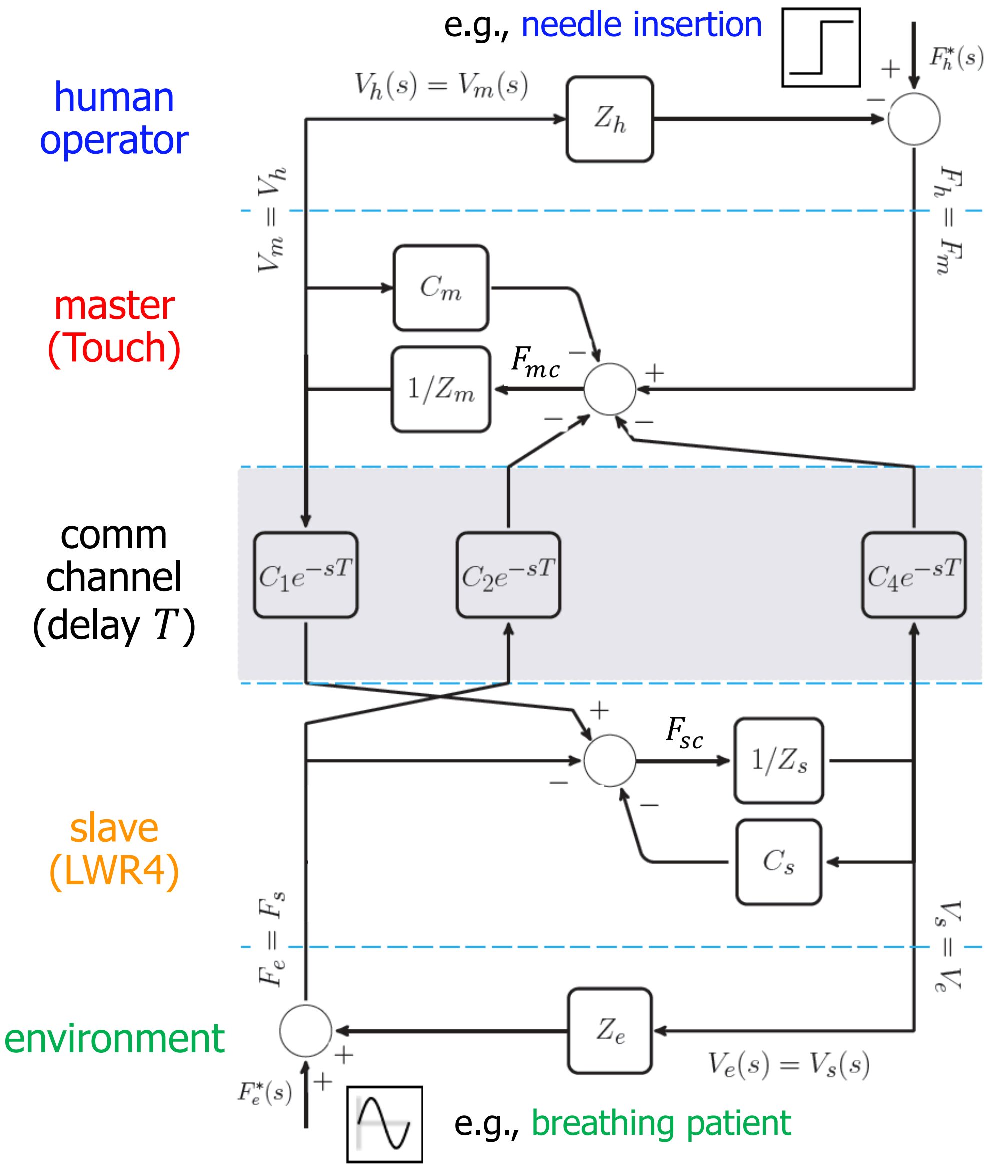


needle insertion in abdominal phantom

reflected **force** on device



Needle insertion steering
3-way teleoperation control scheme



control blocks

$$C_m = B_m + \frac{K_m}{s} \qquad C_s = B_s + \frac{K_s}{s}$$

$$C_1 = C_s \qquad C_2 = K_f \qquad C_3 \text{ not used} \qquad C_4 = -C_m$$

$$F_{mc} = -K_f F_e + \left(B_m + \frac{K_m}{s}\right)(V_e - V_h)$$

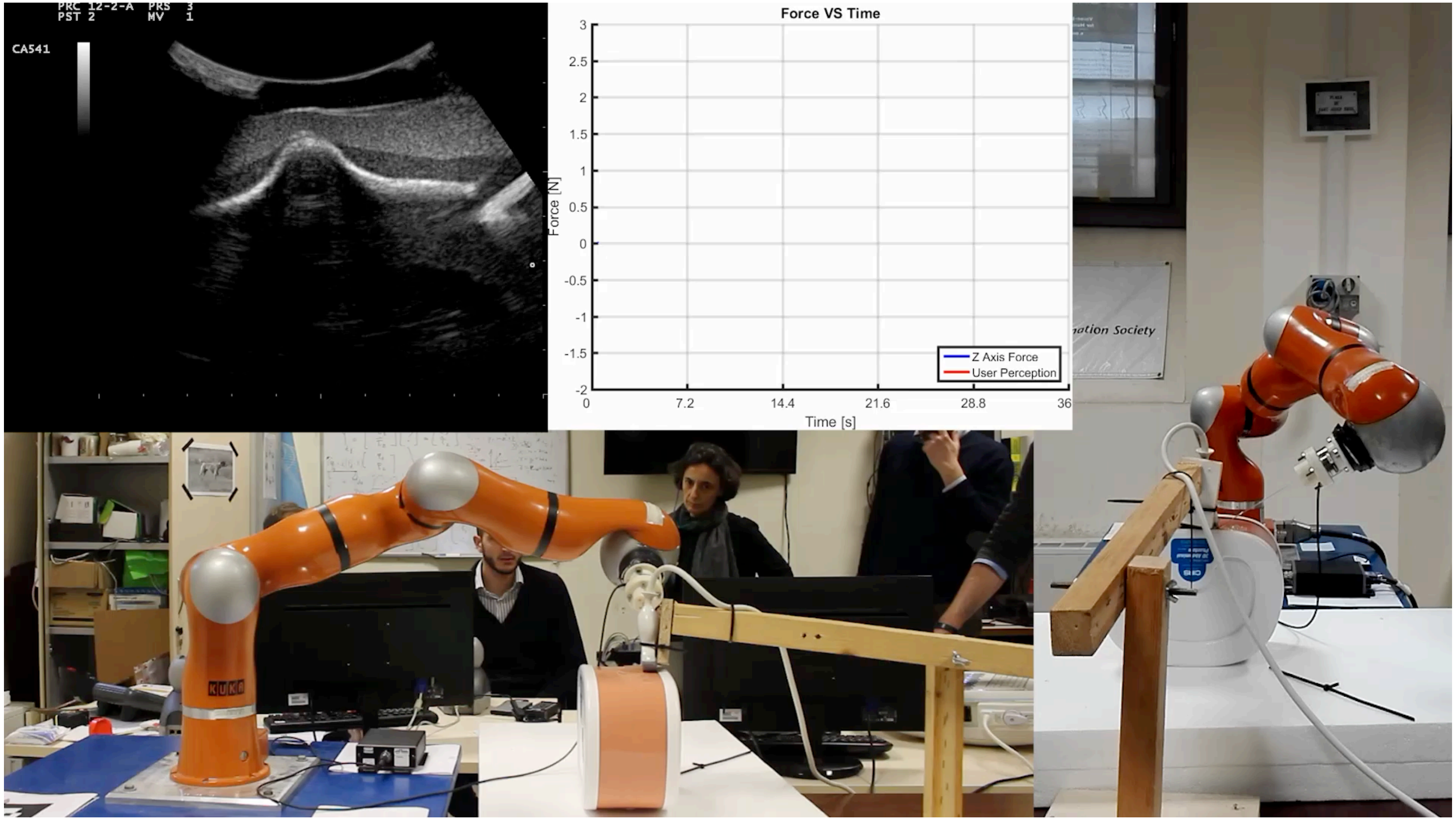
$$F_{sc} = \left(B_s + \frac{K_s}{s}\right)(V_h - V_e)$$

slave robot KUKA LWR has position control loops that are accurate, fast and reliable

$$V_{sc} = \mathbb{Y}_s \left[\left(B_s + \frac{K_s}{s}\right)(V_h - V_e)\right] = \mathbb{Y}_s F_{sc}$$

admittance control of KUKA LWR robot

Needle insertion steering
teleoperation with Geomagic Touch & KUKA LWR4

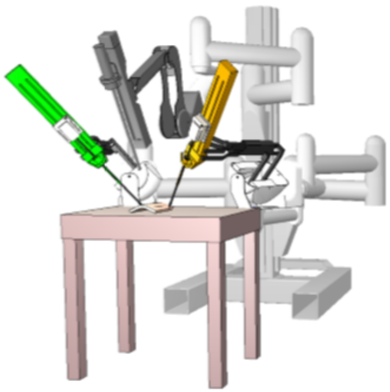


video

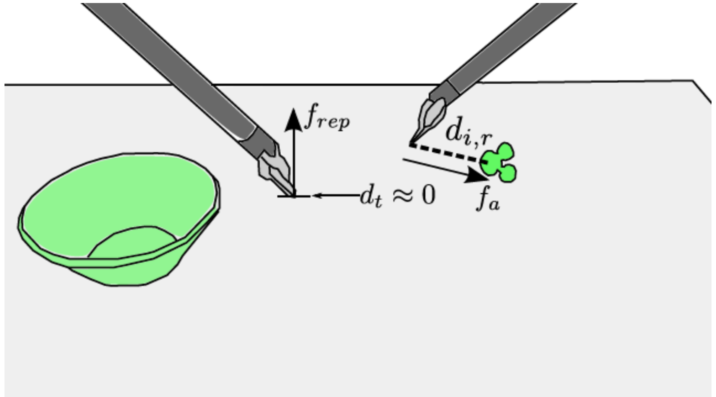
radiologist experiences transitions and bone contact on abdominal phantom

Robotic surgery
portable daVinci simulator

- daVinci (dVRK) patient-cart modelled in V-REP with full kinematics
- 2 Geomagic Touch to emulate the daVinci Master Tool Manipulators
- teleoperation of the Patient Side Manipulators for a pick & place task
  - stylus top button = clutch system to handle kinematic dissimilarity
  - stylus bottom button = grasp/release function
- haptic guidance
  - attractive force towards target to ease grasping
  - repulsive force simulating contact of grippers with table surface



2 Touch + Oculus Rift HMD



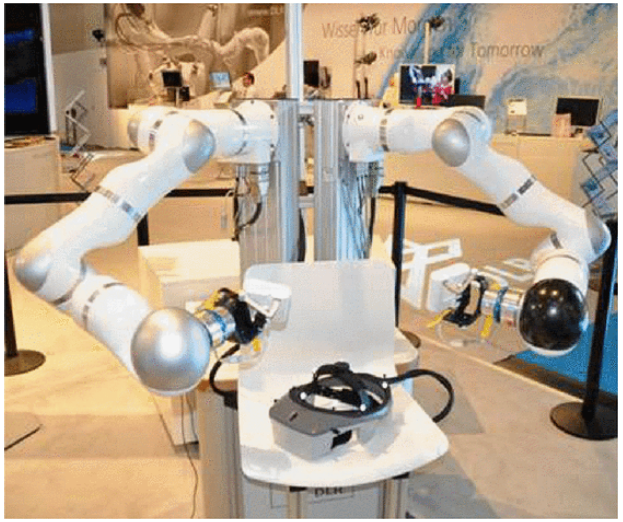
Robotic surgery
portable daVinci Simulator

video

M. Ferro, D. Brunori, F. Magistri, L. Saiella, M. Selvaggio, G.A. Fontanelli, "*A portable da Vinci simulator in virtual reality*", 3rd IEEE International Conference on Robotic Computing, Naples, 2019, pp. 447-448

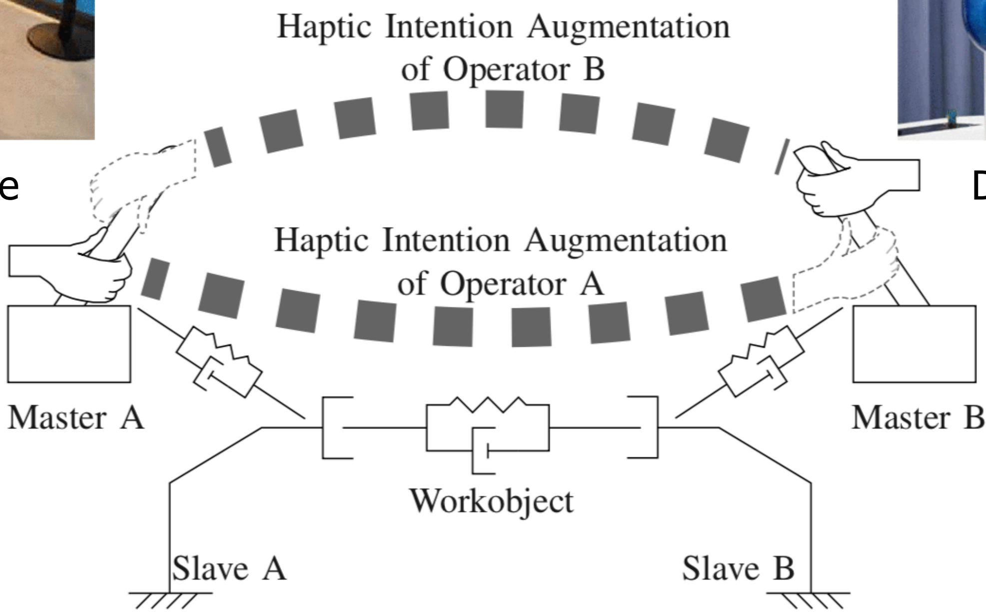
developed in our Sapienza Robotics Lab – presented at Maker Faire in 2018 and 2019

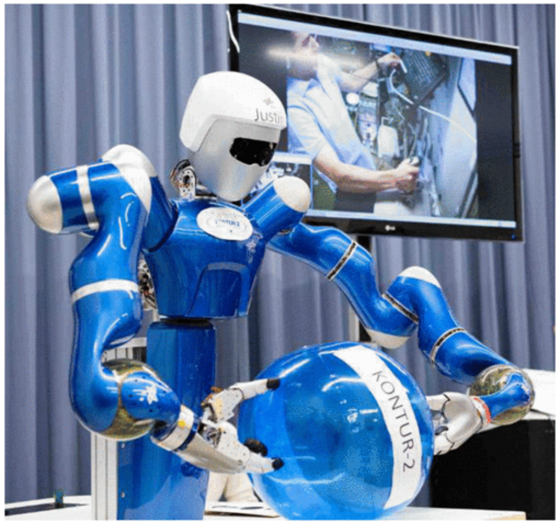
Cooperative teleoperation
e.g. in space applications



DLR HUG Haptic Input Device

2 human operators for
2 remote manipulators
cooperating in a task





DLR Bimanual Space Justin

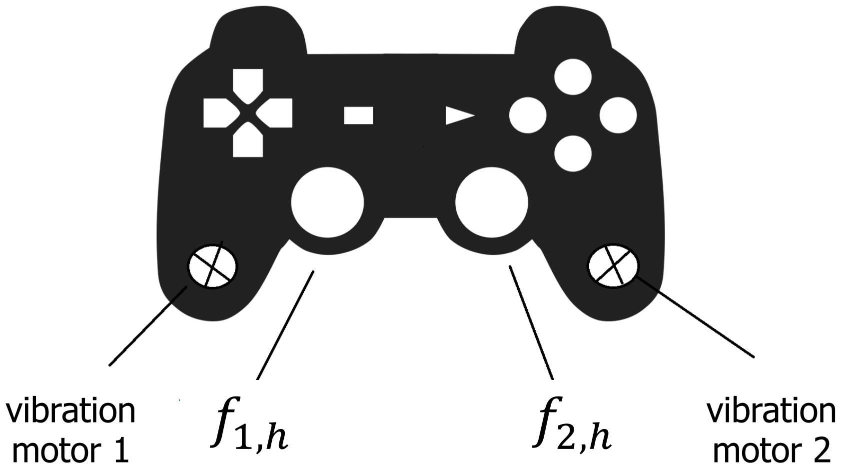
J. Artigas *et al.*,
IEEE ICRA 2016;
M. Panzirsch *et al.*,
IEEE ICRA 2017

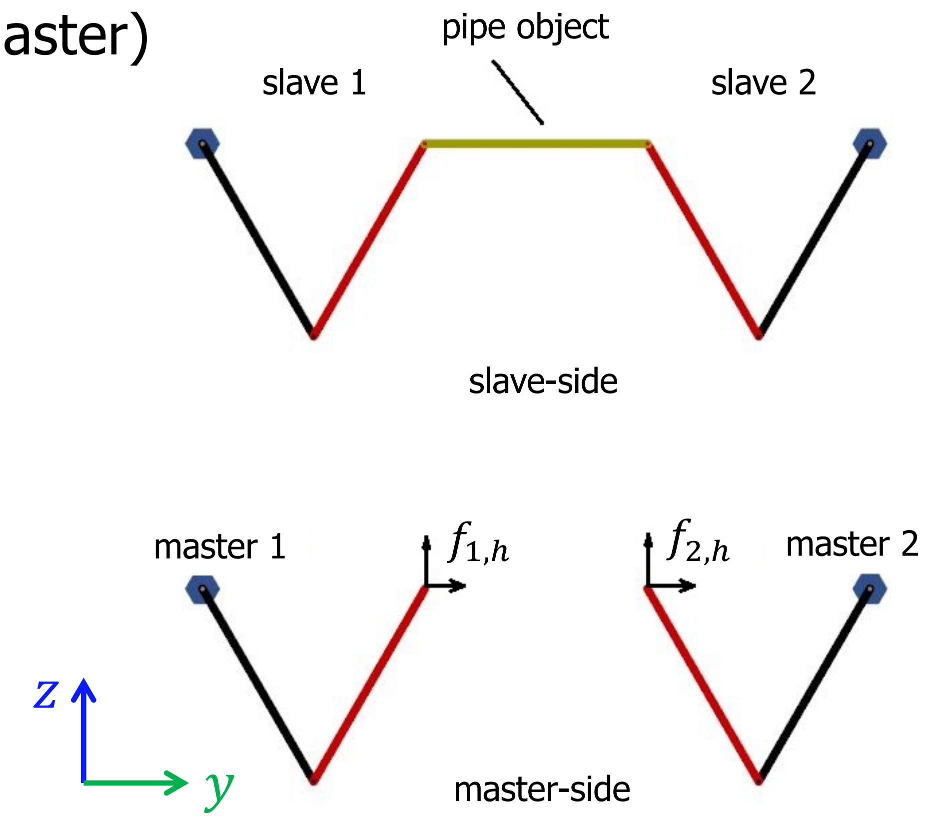
force feedback to each operator enhanced by info on motion intention of the other,
as observed by force sensors mounted at the input devices

Cooperative teleoperation
with haptic augmentation

Haptic intention augmentation for cooperative teleoperation

- multi-robot cooperation tasks enhanced by haptic feedback
- Simulink/Matlab simulation setup
- two 2R planar robots holding a stiff pipe (3-dim planar task)
- commands sent by a joystick (master)
- works also with flexible object

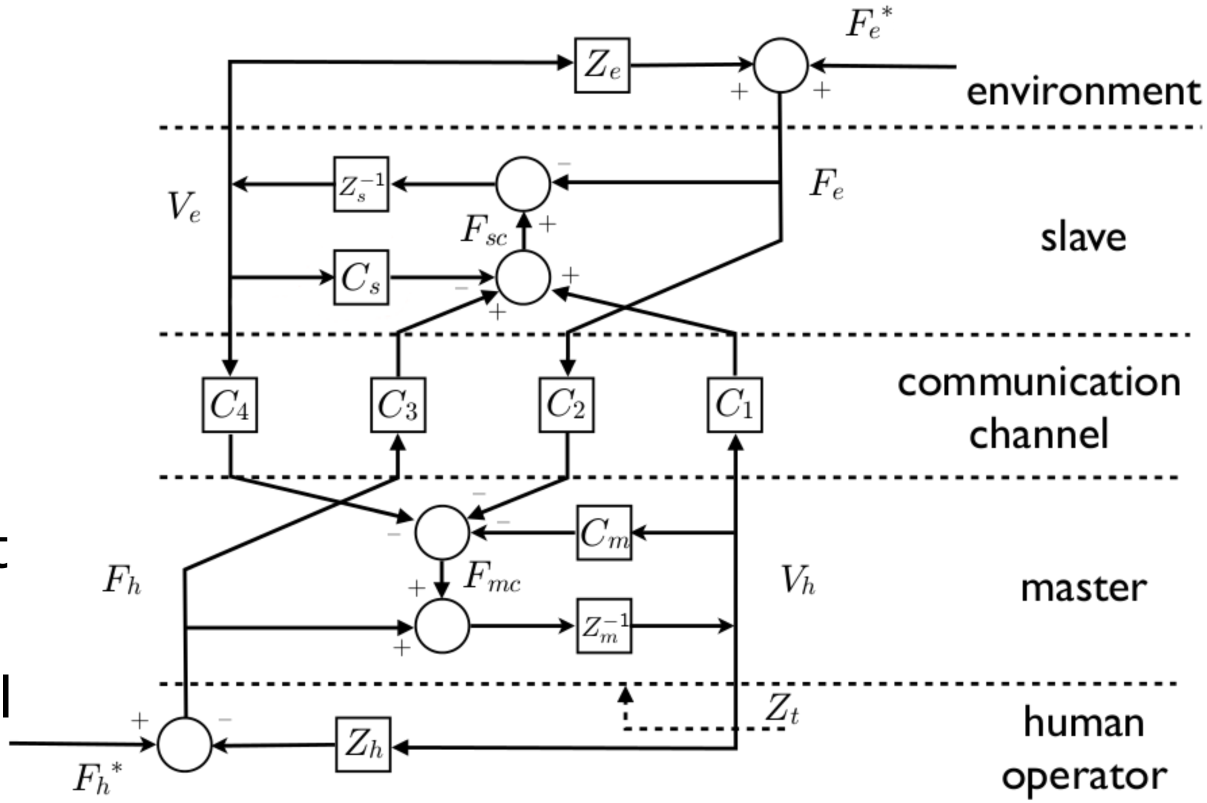




Cooperative teleoperation

control using the (4-way) general scheme

- two 4-way control architectures used for haptic augmentation characterized by
  - two forward channels
    - applied human force forwarded to the slave
    - master position/velocity sent to the slave
  - two feedback channels
    - measured environment force back to master
    - computed slave control force back to master

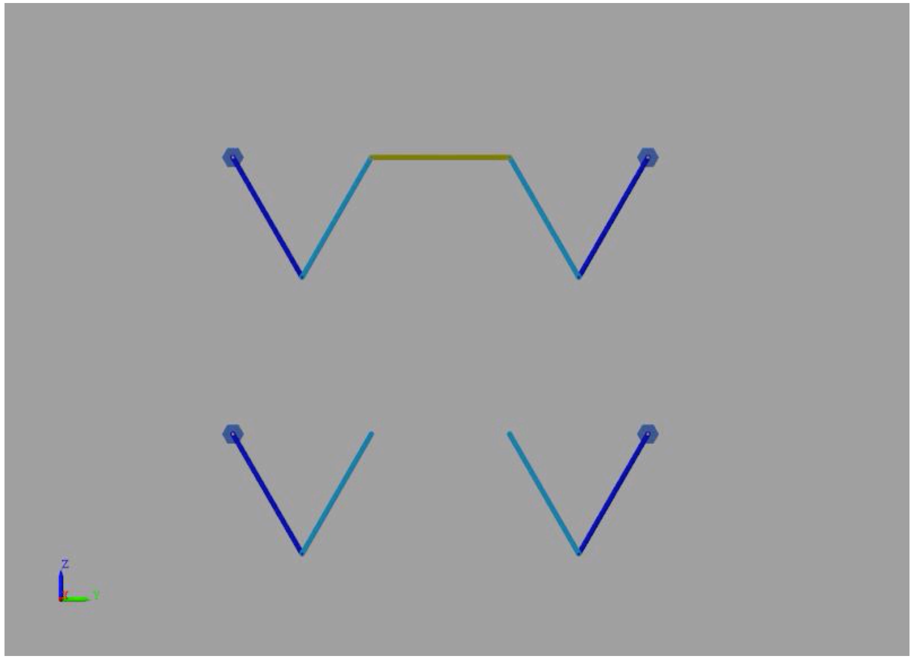


- haptic motion intentions of the users and environment forces are fed back to the master devices via vibrations

$$f_{1mc,FB} = G_1 f_{1e,z} + G_{12} f_{2h,z} \qquad f_{2mc,FB} = G_2 f_{2e,z} + G_{21} f_{1h,z}$$

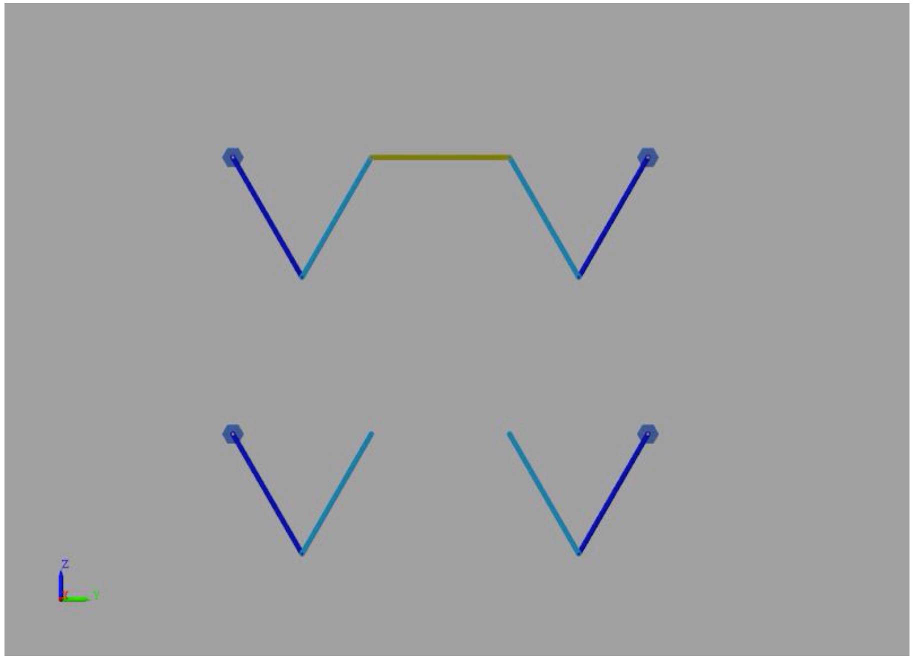
Cooperative teleoperation
comparison on a rigid object

no augmentation



video

with haptic augmentation



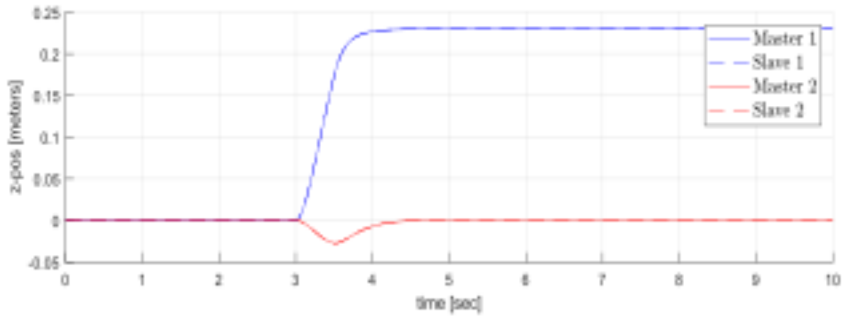
video

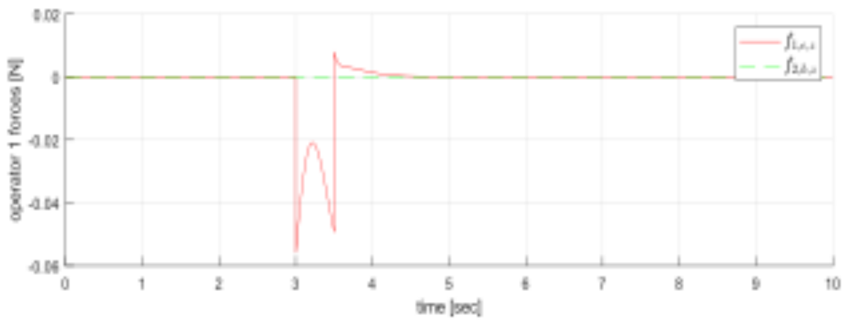
when human operator 1 applies a force in the z-direction for a small time; operator 2 does not apply forces nor resists motion of the second master device

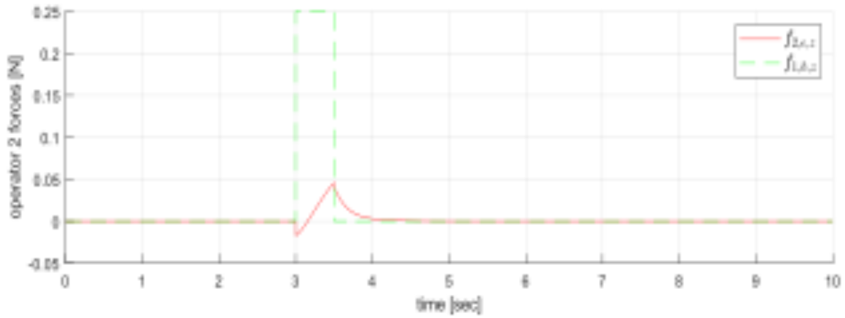
as operator 1 applies force on master 1, this is feedforwarded to master 2, providing the motion intention to operator 2; operator 2 does not resist this force, and master 2 (and so slave 2) starts moving in the same direction instantly

Cooperative teleoperation
comparison on a rigid object

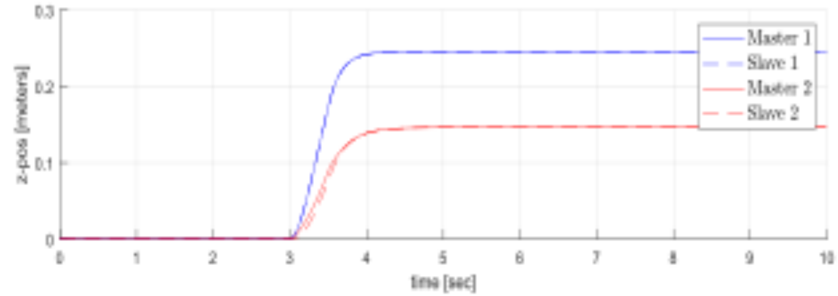
no augmentation

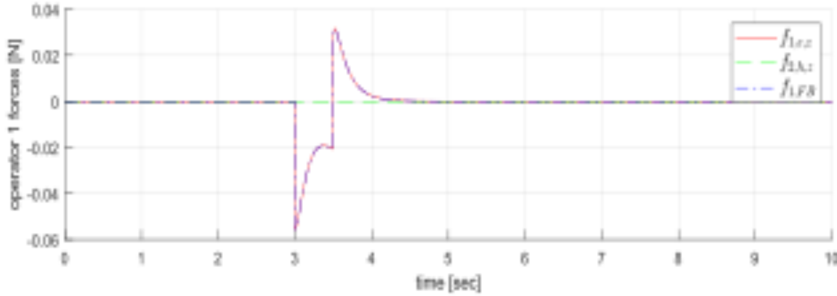


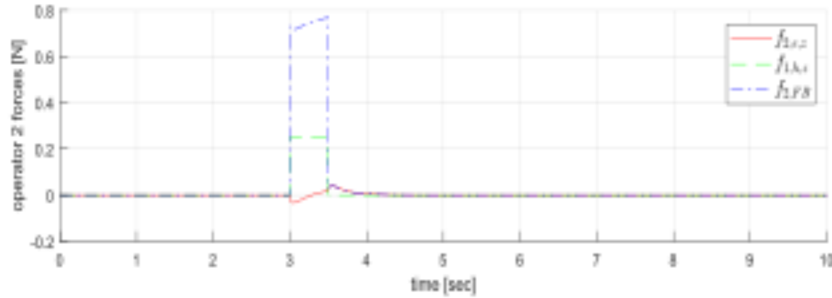




with haptic augmentation

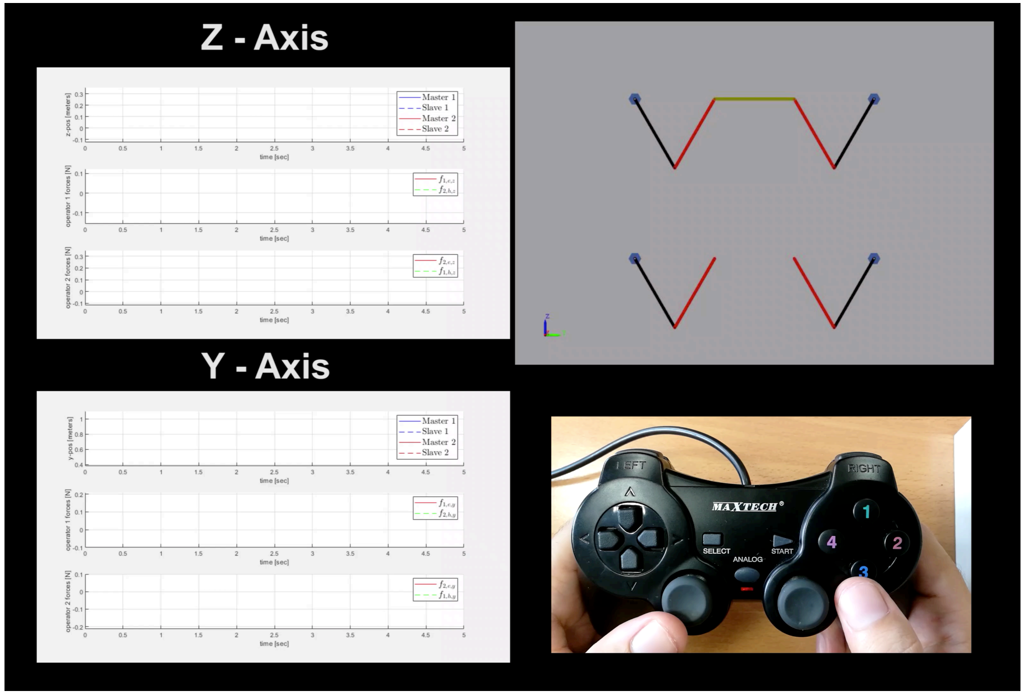




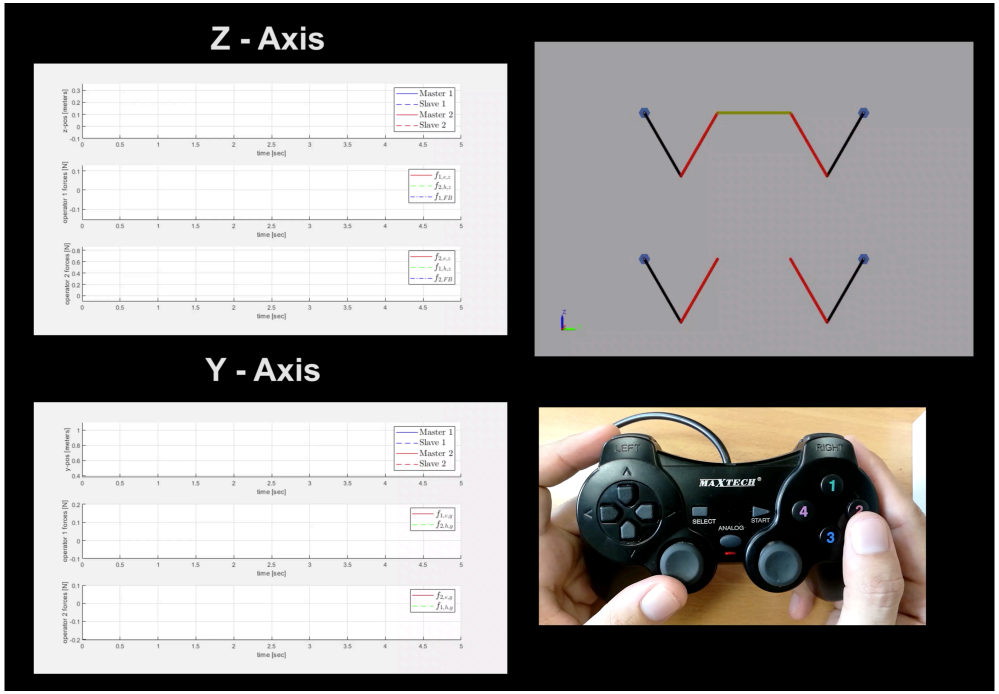


Cooperative teleoperation
comparative live demos

no augmentation



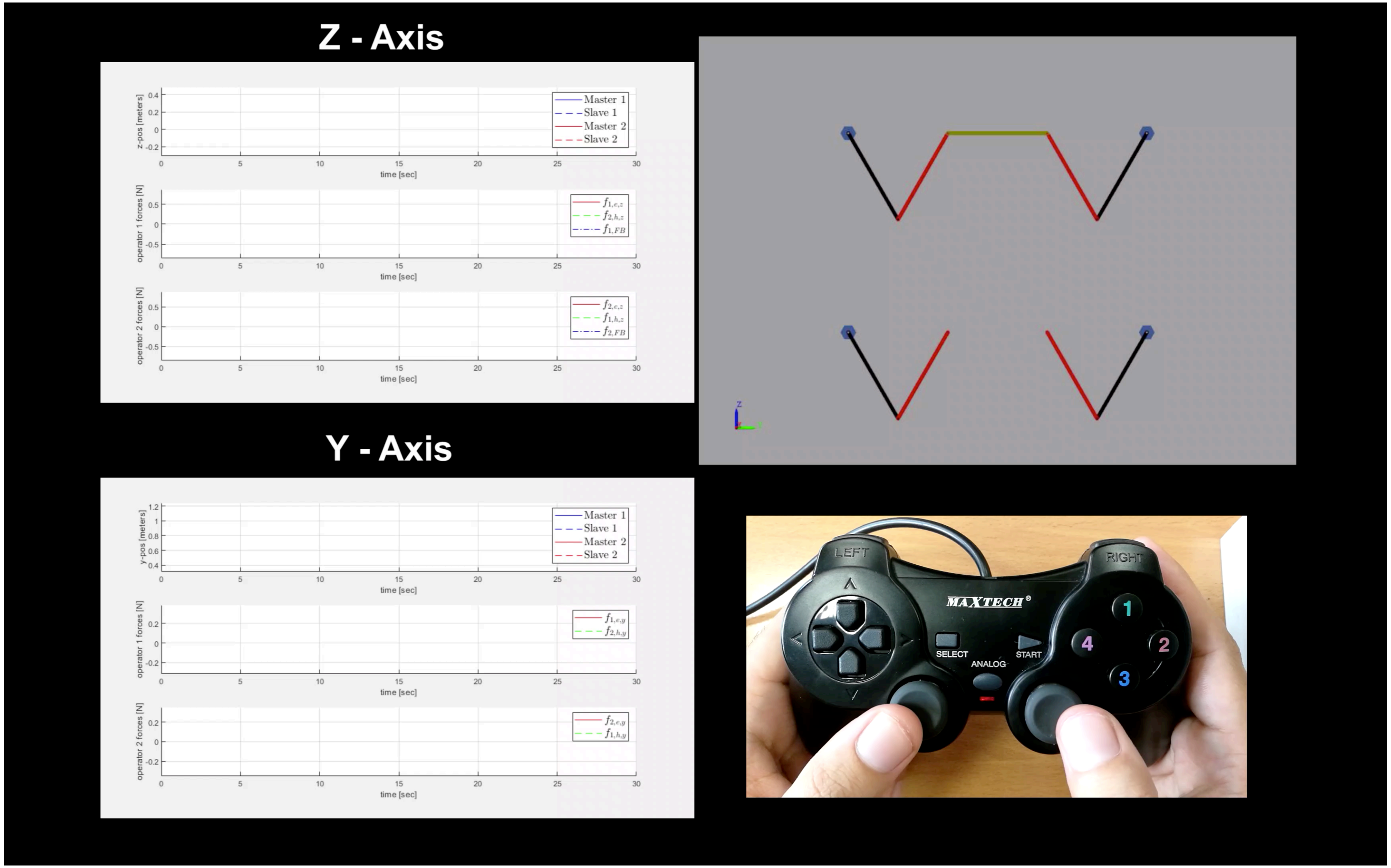
with haptic augmentation



video

video

Cooperative teleoperation
final live demo with haptic augmentation



video

Bibliography

- B. Hannaford, A.M. Okamura, "Haptics," *Springer Handbook of Robotics* (O. Khatib, B. Siciliano, Eds.), Springer, 2nd Ed, pp. 1063-1083, 2016
- K. Salisbury, F. Conti, F. Barbagli, "Haptic rendering: Introductory concepts," *IEEE Computer Graphics and Applications*, vol. 24, pp. 24-32, 2004

- A. Perica, F. Iodice, W. Villa, "Haptic rendering with the Geomagic Touch," LHI short project 2015-16
- D. Evangelista, F. Iodice, L. Monorchio, A. Perica, "Remote needle insertion with reconstructed force feedback," MedRob/Rob2 project 2014-15
- I. Aloise, M. Colosi, A. Gigli, "Simulation framework for the teleoperation of a virtual robot performing needle insertion," MedRob/Rob2 project 2015-16
- G. Borrello, D. De Cillis, R. Germanà, C. Lacerra, "Remote needle insertion steering," MedRob/Rob2 project 2015-16
- A. Afifi, A. Paoli, Y. Abou Dorra, "Haptic intention augmentation for cooperative teleoperation," LHI short project 2017-18