Tasks Specification Framework for Robotic Tasks

Shehzad Ahmed
Course : Mobile Manipulation & Kuka Innvation award
Presentation :Final Exam
Hochschule Bonn-Rhein-Sieg

Introduction

- Mostly robotic tasks require relative motions and/or controlled dynamic interaction between objects.[1]
- Robotic tasks:
  - Simple task:Motion from some initial position to goal position.
  - complex task:Interaction with the objects in the environment.

Task specification

- Task execution require description of task in an intuitive way.
- Tasks can be specified at different levels[2]
  - Abstract level or Motion Planning level
    - Higher level commands e.g. sequence of subtasks/actions which involve one particular relative motion.
  - Discrete level or Robot Control level
    - Defining all the details needed to execute the subtasks task e.g. set points,trajectory,velocities or forces.

Task Specification Framework(TSF)

- Need of generic and systematic framework to:
  - Specify and control complex tasks at abstract level[1] and
  - then transformed automatically to low level control commands respectively.[1]
- In state of the art two such task specification framework has been presented.

TSF-State of the Art

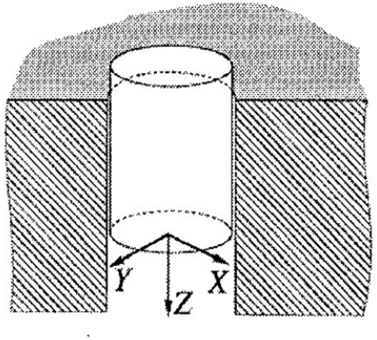
- Compliance Frame Formalism[3]
- Task Frame Formalism[1]
- Constraint based Task specification Formalism[4]

Compliance Frame Formalism

- The very first task formalism was introduced by Mason in [3] for compliant robot motions when manipulator position/motion is constrained by the task geometry.
- He introduced basic concepts to describe the compliant motion tasks based on models of manipulator,task geometry and desired behavior.
- Formalism served as simple separation interface between programmer and manipulator control.

Task Frame Formalism(TFF)

- Bruyninckx and De Schutter (1996) in [5] formally defined the Mason's formalism as Task frame and Task Frame Formalism.
- TFF integrates three important aspects for task specification.
- Modeling as a Task frame:
  - It models the instantaneous contact situations by means of an orthogonal Task Frame(TF).
  - TF has 6 programmable directions along(axial vector) and around(polar vector) three orthogonal axis.
  - TFF models these 6 directions either as position/velocity controlled or as force/moment control.



Task Frame Formalism(TFF)

  - According to mason's TF direction model the so called natural constraints.
- Action specifications:
  - Elementary TFF action is specified by giving desired velocity or force set points along individual 6 TF directions.
  - These task specifications,also called Artificial constraints, must be compatible with modeled TF directions.

Task Frame Formalism(TFF)

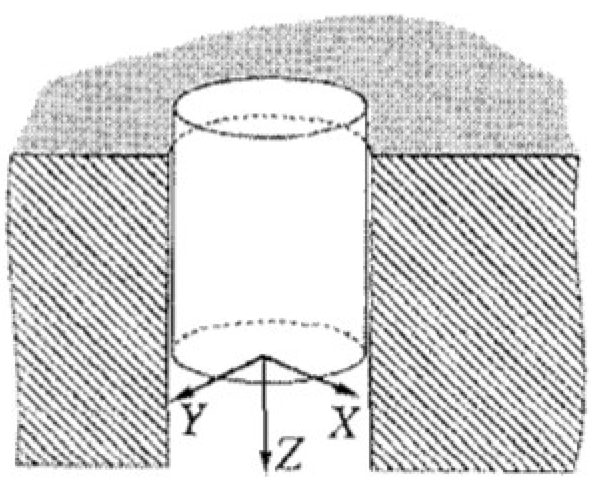
- Adaptation:
  - TFF should keep adapting the position and orientation of TF due to task geometry changes during contact task execution.
  - Adaptation enables two properties:
    - The force and velocity controlled directions do not change.
    - The task specification can use constant motion or force set points.

Task Frame Formalism(TFF)

- These three aspects give rise to two important requirements.
  - Geometrical compatibility
    - TFF should model motion constraints completely.
  - Causal compatibility
    - TF action specification must be compatible with TF constraint model.
  - Time-invariance ensures adaptability.

Task Frame Formalism(TFF)

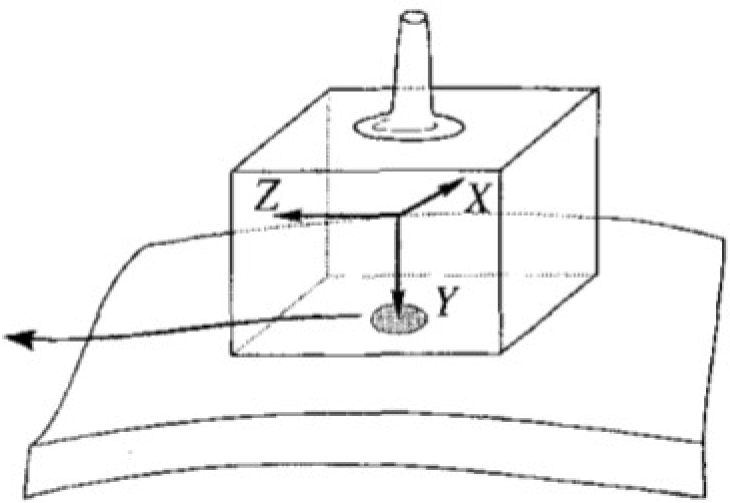
- TFF relies on the robustness of underlying elementary task execution controller if the constraint motion model has uncertainties.
- Task specification of Peg in hole.




```
move compliantly {
    with task frame directions
        xt: force 0 N
        yt: force 0 N
        zt: velocity v mm/sec
        axt: force 0 Nmm
        ayt: force 0 Nmm
        azt: velocity 0 rad/sec
} until zt force < -f N
```

TFF Examples

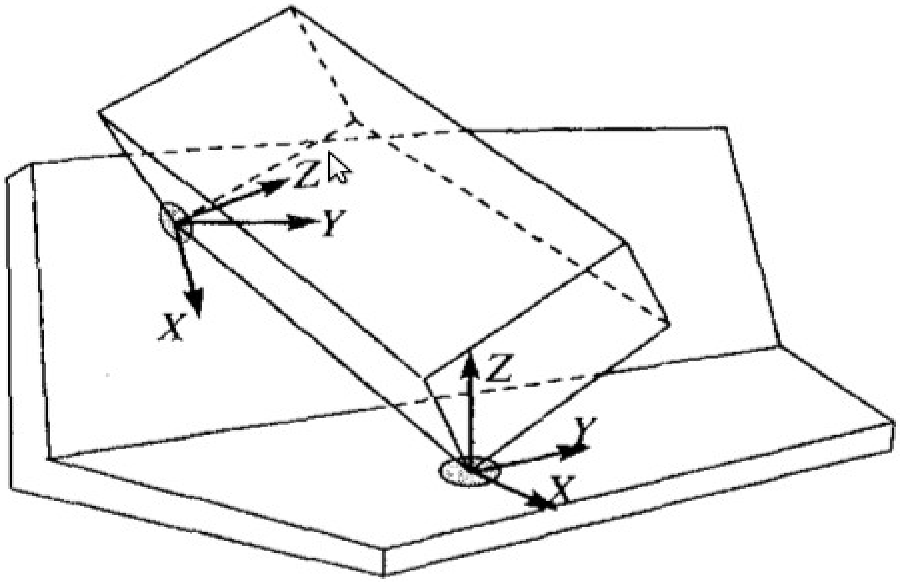
- Sliding a block over a surface



move compliantly {
with task frame directions
xt: velocity v_x mm/sec
yt: force -f N
zt: velocity -v_z mm/sec
axt: force 0 Nmm
ayt: velocity 0 rad/sec
azt: force 0 Nmm
} until time > t sec

TFF limitation

- Bruyninckx and De Schutter (1996) also pointed out the limitation of TFF.
  - It only applies to task geometries with limited complexity.



Constraint based Task specification Formalism

- De Schutter in [5] introduced this formalism to deal with the specification of sensor based complex tasks and geometric uncertainties simultaneously.
- Proposed approach assigns different control modes with corresponding constraints along arbitrary directions in 6D manipulation space.
- Inspired by the work of [6] in constraint based programming,replaced TF and extended to multiple feature frames which enable:
  - Modeling of part of task geometries
  - Specification of part of constraints in each feature frames.

Constraint based Task specification Formalism

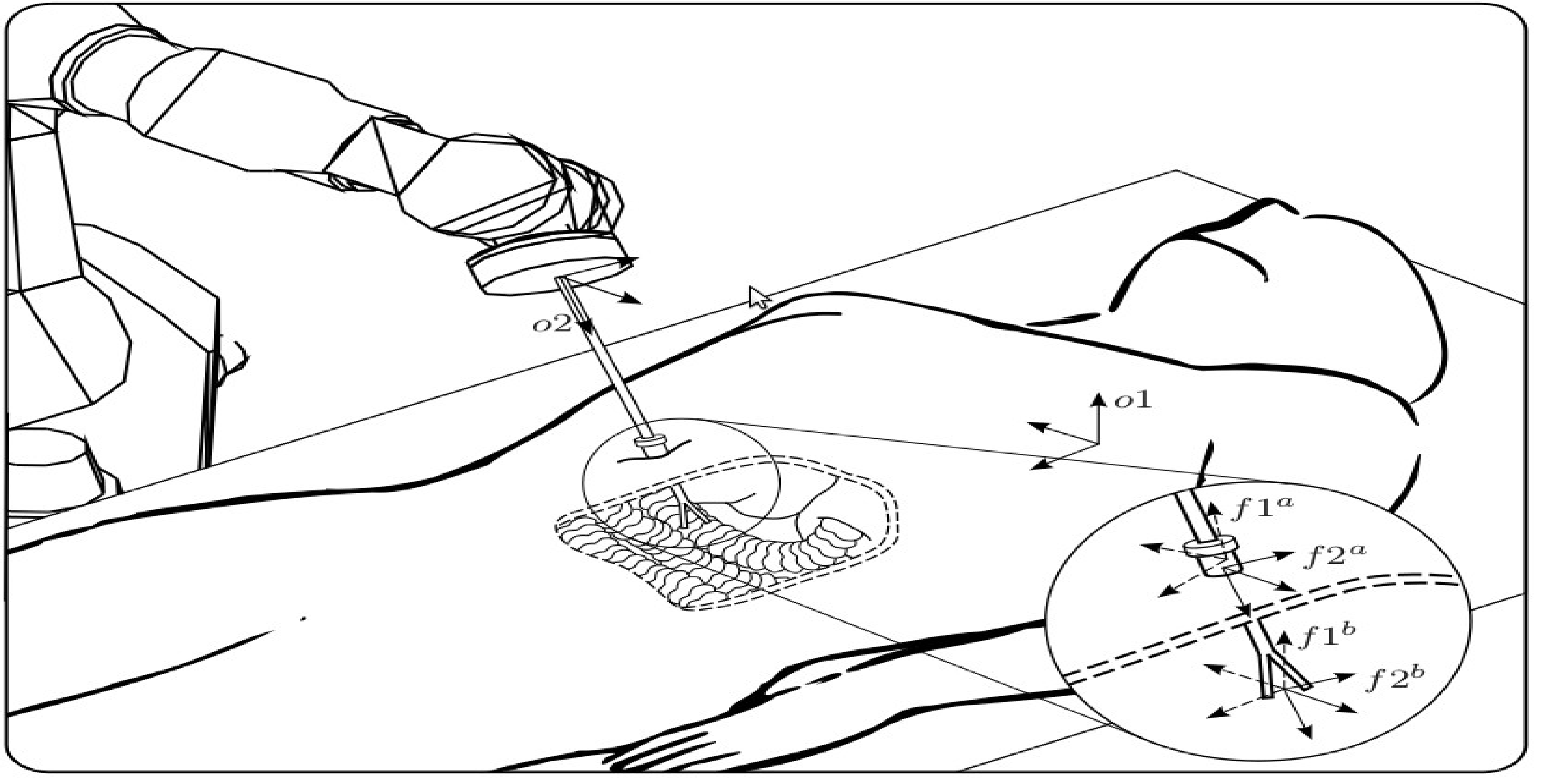
- Also inspired by the work in [7],he proposed to specify the task by imposing constraints on the modeled relative motions and dynamic interactions,called task function approach or constraint based task programming.
- He also introduced the set of uncertainty coordinates to account for geometric uncertainty due to:
  - Modeling errors,uncontrolled DOF or geometric disturbances.
  - Inclusion of uncertainty coordinates as states in robot dynamic model of robot system.

Constraint based Task specification Formalism

- A velocity based resolved control law is proposed to link constraint based motion specification to real time task execution.

Object and Feature frames

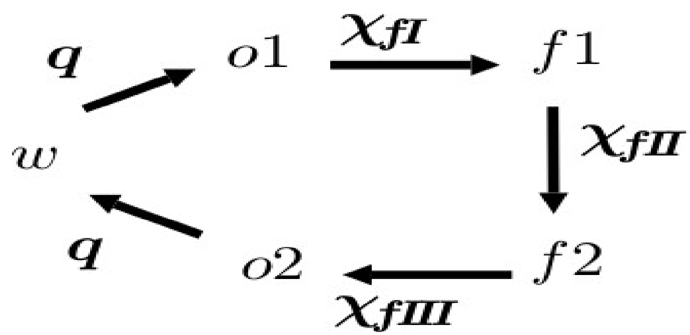
- Minimal invasive surgery example:



- Identification of object(end-effector) and features(physical entity S.A vertex,edge,surface) that are relevant for task.
- Imposing constraint on relative motion or force between one feature on first object and a corresponding feature on the second object.
- Feature coordinates(Xf):
  - Every feature sub-motion can be represented by minimal set of position coordinates which combines to give to feature coordinates.

    dim(Xf) = 6*nf ,where nf represens feature relationship

- The surgery task distributes 6DOF between o1 and o2 as follows:



Object and feature frames and feature coordinates.

For feature a:

$$
\begin{aligned}
{\chi_{fI}}^a &= (-),\\
{\chi_{fII}}^a &= \begin{pmatrix} x^a & y^a & \phi^a & \theta^a & \psi^a \end{pmatrix}^T\\
{\chi_{fIII}}^a &= (z^a).
\end{aligned}
$$

For feature b:

$$
\begin{aligned}
{\chi_{fI}}^b &= \begin{pmatrix} x^b & y^b & z^b \end{pmatrix}^T\\
{\chi_{fII}}^b &= \begin{pmatrix} \phi^b & \theta^b & \psi^b \end{pmatrix}^T\\
{\chi_{fIII}}^b &= (-).
\end{aligned}
$$

χ_{fI} represents the relative motion of $f1$ with respect to $o1$,

χ_{fII} represents the relative motion of $f2$ with respect to $f1$, and

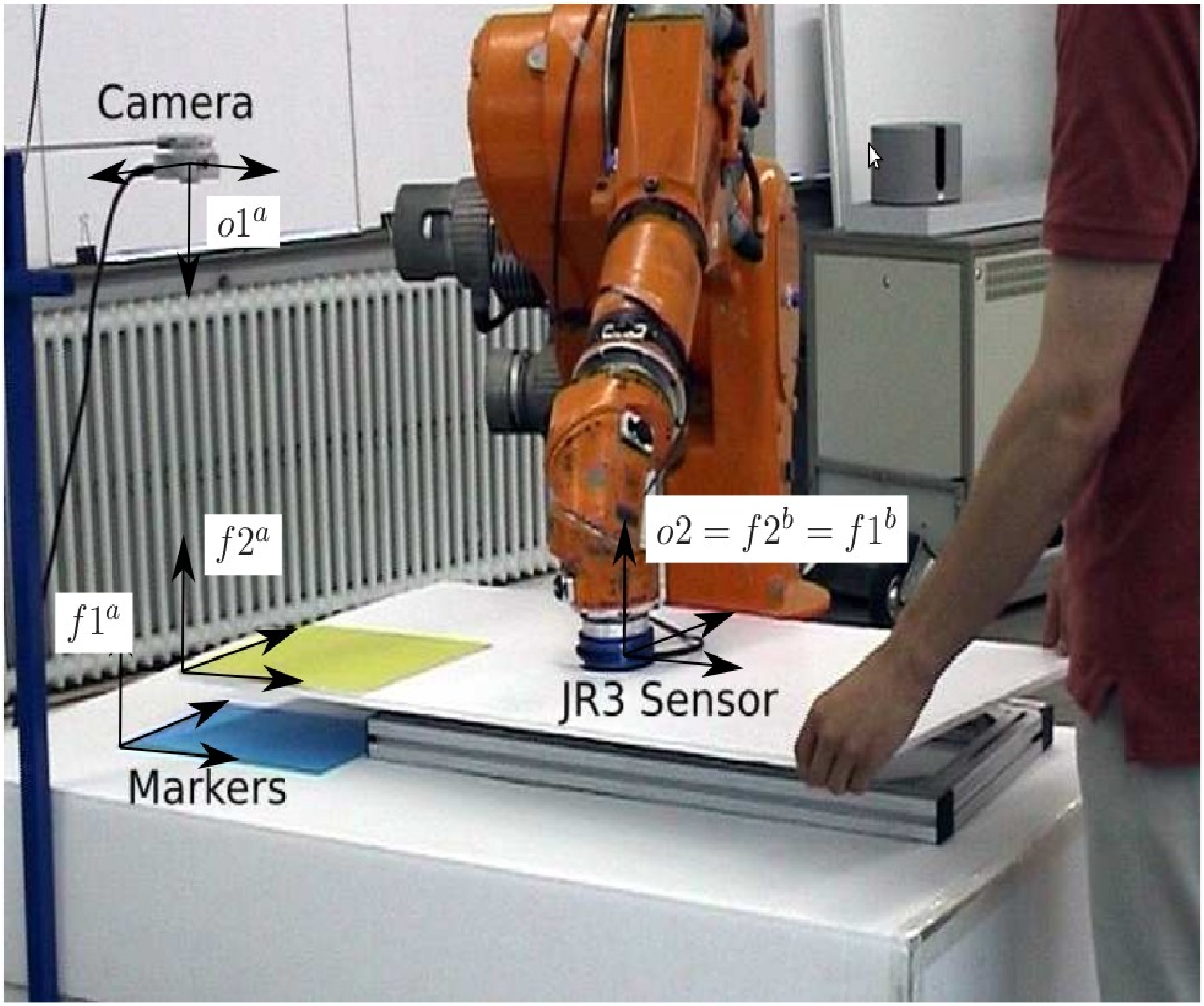
χ_{fIII} represents the relative motion of $o2$ with respect to $f2$.

Uncertainty coordinates

- Two types of geometric uncertainty
  - Uncertainty on the pose of an object
  - Uncertainty on the pose of a feature with respect to its corresponding object

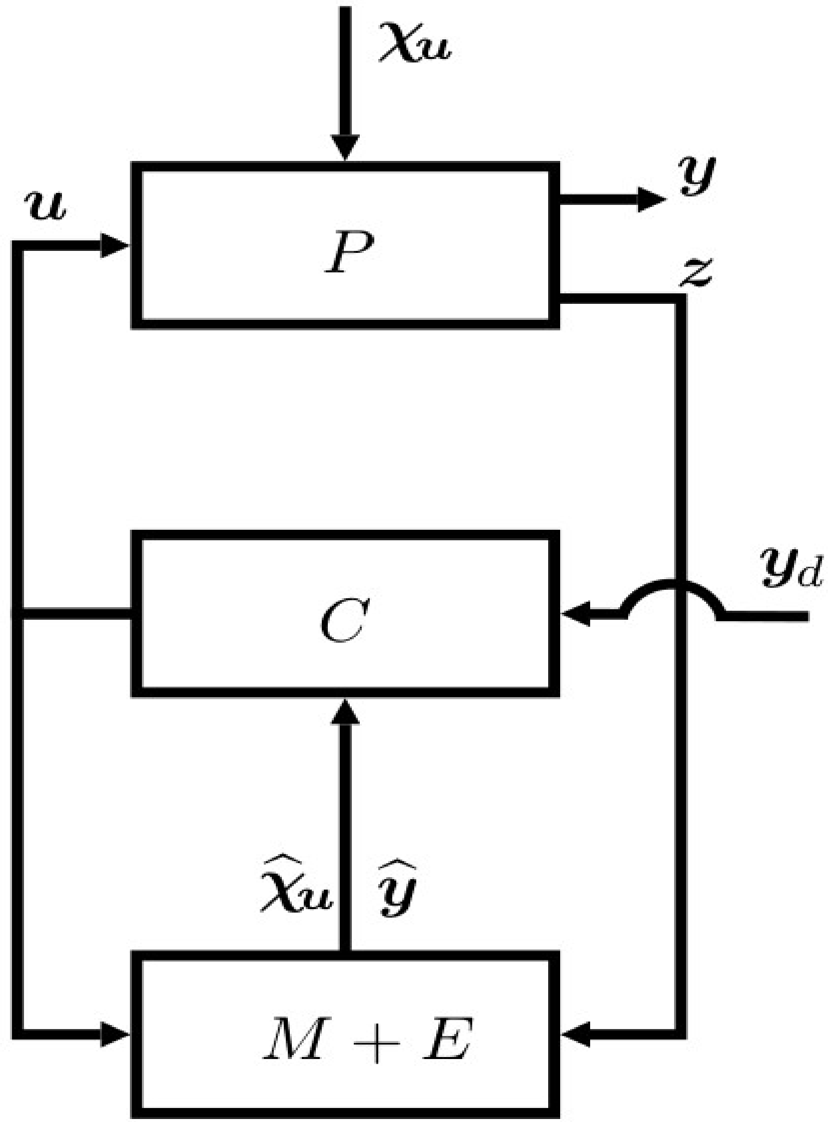
$$
\begin{tikzcd}
 & o1' \arrow[r, "\chi_{uI}"] & o1 \arrow[r, "\chi_{fI}"] & f1' \arrow[r, "\chi_{uII}"] & f1 \arrow[dd, "\chi_{fII}"] \\
w \arrow[ur, "\boldsymbol{q}"] & & & & \\
 & o2' \arrow[ul, "\boldsymbol{q}"] & o2 \arrow[l, "\chi_{uIV}"] & f2' \arrow[l, "\chi_{fIII}"] & f2 \arrow[l, "\chi_{uIII}"]
\end{tikzcd}
$$

- A task is easily specified using the task coordinates χf and χu
- The goal of this task is threefold
  - The tool has to go through the trocar.
  - Three translations between the tool and the patient are specified.
  - Two supplementary rotations may be specified.
- The outputs to be considered for this task are:
- $y_1 = x^a, \quad y_2 = y^a, \quad y_3 = x^b,$
  $y_4 = y^b, \quad y_5 = z^b, \quad y_6 = \phi^b, \text{ and } \quad y_7 = \theta^b.$



The experimental setup for the human-robot co-manipulation task.

General control and estimation scheme



Conclusion

- TFF uses on TF to describe the task but it can only with tasks with simple geometric constraints.

- Constraint based formalism uses multiple features and feature coordinates to simplify the description of complex tasks.
- Moreover it also includes estimation of geometric uncertainities.

iTaSC framework

- Based on the constraint based task formalism, a software implementation of this frame is also been provided which is called iTaSC (instantaneous Task Specification using Constraints)
- This framework is to generate robot motions by specifying constraints between (parts of) the robots and their environment.
- iTaSC was born as a specification formalisms to generalize and extend existing approaches, such as the Operational Space Approach, the Task Function Approach, the Task Frame Formalism, geometric Cartesian Space control, and Joint Space control.
- The iTaSC concepts is also extended to include equality and inequality constraints. [8]

Key advantages of iTaSC over traditional motion specification[9]

- Composability of partial constraints
- Reusability of constraint specification
- Automatic derivation of the control solution
- Weights and priorities

References

[1] Specification of Force-Controlled Actions in the “Task Frame Formalism”-A Synthesis

[2] Robot task specification and execution through relational positioning

[3] Compliance and Force Control for Computer Controlled Manipulators

[4] Constraint-Based Task Specification and Estimation for Sensor Based Robot Systems in the Presence of Geometric Uncertainty.

[5] A framework for compliant physical interaction

[6] A. P. Ambler and R. J. Popplestone. Inferring the positions of bodies from specified spatial relationships Artificial Intelligence, 6:157–174, 1975.

[7] C. Samson, M. Le Borgne, and B. Espiau. Robot Control, the Task Function Approach. Clarendon Press,Oxford, England, 1991.

[8] W. Decre, R. Smits, H. Bruyninckx, and J. De Schutter. Extending iTaSC to support inequality constraints and non-instantaneous task specification. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, pages 964–971, Kobe, Japan, 2009.

[9]http://www.orocos.org/wiki/orocos/itasc-wiki/1-what-itasc