Constrained Robot Motion Control and “Virtual Fixtures”

Russell H. Taylor
600.445
Surgical Assistant Systems

- Situation assessment
- Task strategy & decisions
- Sensory-motor coordination

Augmentation System
- Sensor processing
- Model interpretation
- Display
- Online references & decision support
- Manipulation enhancement
- Cooperative control and "macros"

Surgical Assistant Systems

- Situation assessment
- Task strategy & decisions
- Sensory-motor coordination

- Sensor processing
- Model interpretation
- Display
- Online references & decision support
- Manipulation enhancement
- Cooperative control and "macros"
Surgical Assistant Systems

- Situation assessment
- Task strategy & decisions
- Sensory-motor coordination

Augmentation System
- Sensor processing
- Model interpretation
- Display
- Online references & decision support
- Manipulation enhancement
- Cooperative control and "macros"

Problem: specifying motion for a [medical] robot

Task level control
- Desired motion description
- Task - level constraints on how motion is done
- Surgeon input
  - Plan information
  - Anatomic models
  - Safety constraints

Motion level control
- Joint position or velocity commands
- Sensor and state information
- Robot kinematics & motion limits

Low level control
- Motor currents
- Sensor values
- Joint positions & velocities
- Joint positions, velocities, & other state information
Background: Jacobean Robot Motion Control

Let \( \mathbf{F} = [\mathbf{R}, \mathbf{p}] \) be the current pose of a robot end effector and
\( \mathbf{q} = [q_1, \ldots, q_n] \) be the current joint position values corresponding
to \( \mathbf{F} \). I.e., \( \mathbf{F} = \text{Kins}(\mathbf{q}) \), where \( \text{Kins}(\cdots) \) is a function computing
the "forward kinematics" of the robot.

\[
\text{Joint positions } \mathbf{q} \quad \text{Pose } \mathbf{F} = \text{kins}(\mathbf{q})
\]

\[
\text{Pose } \mathbf{F}(\mathbf{q} + \Delta \mathbf{q}) = \text{kins}(\mathbf{q} + \Delta \mathbf{q})
\]
\[
\Delta \mathbf{F} \cdot \mathbf{F} = \text{kins}(\mathbf{q} + \Delta \mathbf{q})
\]
\[
\Delta \mathbf{F} = \text{kins}(\mathbf{q} + \Delta \mathbf{q}) \text{kins}(\mathbf{q})^{-1}
\]

For small \( \Delta \mathbf{q} \), we can write the following expression for \( \Delta \mathbf{F} = [\text{Rot}(\vec{a}), \vec{e}] \)

\[
\Delta \mathbf{F} = \text{Kins}(\mathbf{q} + \Delta \mathbf{q}) \text{Kins}(\mathbf{q})^{-1}
\]

which we typically linearize as

\[
\Delta \mathbf{x} = \begin{bmatrix} \vec{a} \\ \vec{e} \end{bmatrix} = \mathbf{J}_{\text{kins}}(\mathbf{q}) \Delta \mathbf{q}
\]

Note that here we are computing \( \Delta \mathbf{F} \) in the base frame of the robot.
If we want to compute \( \Delta \mathbf{F} \) in the end effector frame, so that
\( \mathbf{F} \cdot \Delta \mathbf{F} = \text{Kins}(\mathbf{q} + \Delta \mathbf{q}) \), then we will get a slightly different expression
for \( \mathbf{J}_{\text{kins}}(\mathbf{q}) \), though the flavor will be the same.
Background: Jacobean Robot Motion Control

\[ \text{Pose } F(q + \Delta q) = \text{kins}(q + \Delta q) \]
\[ \Delta F \cdot \Delta q = \text{kins}(q + \Delta q) \]
\[ \Delta F = \text{kins}(q + \Delta q) \cdot \text{kins}(q)^{-1} \]
\[ \begin{bmatrix} \Delta \dot{q} \\ \Delta q \end{bmatrix} \approx \begin{bmatrix} \dot{q} \\ q \end{bmatrix} \cdot \Delta \dot{q} \]
\[ \Delta \dot{q} \approx \begin{bmatrix} \dot{q} \\ q \end{bmatrix}^{-1} \begin{bmatrix} \Delta \dot{q} \\ \Delta q \end{bmatrix} \]

One implementation

\[ \Delta \dot{x}_{\text{des}} \]
\[ \dot{q}_{\text{des}} = q + J^{-1}(q) \Delta \dot{x}_{\text{des}} \]
\[ \Delta \dot{q} \]
\[ \dot{q} \]
\[ \ddot{q} \]

Sensor values

Robot kinematics & motion limits
\[ \text{Kins}(\cdots), J_{\text{kins}}(\cdots) \]
\[ \dot{q}_L \leq \dot{q} \leq \dot{q}_U \]

Surgeon input
Plan information
Anatomic models
Safety constraints

Motor currents
Steady Hand Robot
Hands on compliance control

\[ \dot{x}_{\text{des}} = K_v f_h \]
\[ q_{\text{cmd}} = J^{-1} \text{kina} \dot{x}_{\text{des}} \]


---

Steady Hand Robot
Hands on compliance control with force scaling

\[ \dot{x}_{\text{des}} = K_v (f_h - \gamma f_{\text{tip}}) \]
\[ q_{\text{cmd}} = J^{-1} \text{kina} \dot{x}_{\text{des}} \]


Example: Fenestratration of Stapes Footplate
Virtual Fixtures

- Bridge the gap between autonomous robots and direct human control.
- Assist the human operator in safer, faster, and more accurate task completion.

Broadly Categorized
- Guidance VF
- Forbidden Region VF
- Different implementation
- Tele-manipulation
- Cooperative Control

Background: Virtual Fixtures

- First proposed for complex telerobotic tasks, but draw upon rich prior research in robot assembly and other manufacturing automation applications

- Many authors, e.g.,

- Discussion that follows draws upon work at IBM Research and within the CISST ERC at JHU. E.g.,
Original Motivation for IBM Work

- Kinematic control of robots for MIS
- E.g., LARS and HISAR robots
- LARS and other IBM robots were kinematically redundant
  - Typically 7-9 actuated joints
- But tasks often imposed kinematic constraints
  - E.g., no lateral motion at trocar
- Some robots (e.g., IBM/JHU HISAR and CMI’s AESOP) had passive joints
- General goals
  - Exploit redundancy in best way possible
  - Come as close as possible to providing desired motion subject to robot and task limits
- **Our approach:** view this as a constrained optimization problem

LARS degrees of freedom

- Video tracking
- Clip-on joystick
- View direction
- XYZ
LARS Video
Motion Specification Problem

• Requirements
  – The tool shaft must pass within a specified distance of the entry port into the patient’s body
  – The individual joint limits may not be exceeded

• Goals
  – Aim the camera as close as possible at a target
    • or move view in direction indicated by clip-on pointing device
    • or move to track a video target on an instrument
    • or aim the working channel of the endoscope at a target
    • or something else (maybe a combination of goals)
  – Keep the view as “upright” as possible
  – Tool should pass as close as possible to entry port center
  – Keep joints far away from their limits, to preserve options for future motion
  – Minimize motion of XYZ joints
  – Etc.

Our approach: view as an optimization problem

• Currently formulate problem as constrained least squares problem
• Express goals in the objective function
• If multiple goals, objective function is a weighted sum of individual elements
• Add constraints for requirements
• Express constraints and objective function terms in whatever coordinate system is convenient
• Use Jacobean formulation to transform to joint space
• Solve for joint motion
Example: keep tool tip near a point

\[ \vec{D}(\vec{x}) = \Delta F(q, \Delta q) \cdot F \cdot \vec{p}_{tp} - \vec{p}_{goal} \]
\[ = \vec{\alpha} \times \vec{t} + \vec{\epsilon} + \vec{t} - \vec{p}_{goal} \quad \text{where} \quad \vec{t} = F \cdot \vec{p}_{tp} \]
\[ \vec{\alpha} = J_\alpha(q) \Delta q \]
\[ \vec{\epsilon} = J_\epsilon(q) \Delta q \]

Suppose we want to stay as close as possible while never going beyond 3mm from goal and also obeying joint limits

\[ \Delta q_{\text{des}} = \arg \min_{\Delta q} \| \vec{D}(\Delta x) \|^2 = \left\| \vec{\alpha} \times \vec{t} + \vec{\epsilon} + \vec{t} - \vec{p}_{goal} \right\|^2 \]

Subject to

\[ \vec{\alpha} = J_\alpha(q) \Delta q \]
\[ \vec{\epsilon} = J_\epsilon(q) \Delta q \]
\[ \| \vec{\alpha} \times \vec{t} + \vec{\epsilon} + \vec{t} - \vec{p}_{goal} \| \leq 3 \]
\[ \bar{q}_l - \bar{q} \leq \Delta \bar{q} \leq \bar{q}_u - \bar{q} \]
Solving the optimization problem

- **Constrained linear least squares**
  - Combine constraints and goals from task and robot control
  - Linearize and constrained least squares problem
    \[ \Delta \mathbf{q}_{\text{obs}} = \arg \min_{\Delta \mathbf{q}} \left\| E_{\text{task}} \Delta \mathbf{x} - \mathbf{f}_{\text{task}} \right\|^2 + \left\| E_{\mathbf{q}} \Delta \mathbf{x} - \mathbf{f}_{\mathbf{q}} \right\|^2 \]
    subject to
    \[ \Delta \mathbf{x} = \mathbf{J} \Delta \mathbf{q}; \quad \mathbf{A}_{\text{task}} \Delta \mathbf{x} \leq \mathbf{b}_{\text{task}}; \quad \mathbf{A}_{\mathbf{q}} \Delta \mathbf{q} \leq \mathbf{b}_{\mathbf{q}} \]
  - E.g., using “non-negative least squares” methods developed by Lawson and Hanson
  - Approach used in our IBM work and in Kumar, Li, Kapoor theses

- **Constrained nonlinear least squares**
  - Approach explored by Kapoor (discuss later)

---

**Linear least squares implementation**

**Task level control**
- \( E_{\text{task}}, \mathbf{f}_{\text{task}} \)
- \( \mathbf{A}_{\text{task}}, \mathbf{b}_{\text{task}} \)
- Sensor values \( \mathbf{F}, \mathbf{q} \)

**Motion level control**
- \( \Delta \mathbf{q}_{\text{obs}} = \arg \min_{\Delta \mathbf{q}} \left\| E_{\mathbf{x}} \Delta \mathbf{x} - \mathbf{f}_{\mathbf{x}} \right\|^2 \)
- subject to
  \[ \Delta \mathbf{x} = \mathbf{J} \Delta \mathbf{q}; \quad \mathbf{A} \left\| \Delta \mathbf{x}, \Delta \mathbf{q} \right\| \leq \mathbf{b} \]

**Position control**
- \( \Delta \mathbf{q}_{\text{obs}}, \mathbf{q}, \mathbf{q} \)
- Motor currents \( \mathbf{q}, \mathbf{q} \)

**Surgeon input**
- Plan information
- Anatomic models
- Safety constraints

**Robot kinematics & motion limits**
- \( \mathbf{K}_{\text{ins}}(\cdot), \mathbf{J}_{\text{ins}}(\cdot) \)
- \( \mathbf{q}_{\text{c}} \leq \mathbf{q} \leq \mathbf{q}_{\text{c}}; \quad E_{\mathbf{q}} \mathbf{f}_{\mathbf{q}} \)
Some IBM Movies

Early Constrained Motion System (LapSYS)  Vision-guided targeting

Steady Hand Robot
High Level Constrained Control

\[
\text{Optimization Framework}
\begin{align*}
\arg\min_{\dot{q}} & \left\| W(\dot{x} - \dot{q}) \right\|_2^2 \\
\text{s.t.} & \quad H\dot{x} \geq h \\
\dot{x} & = J\dot{q}
\end{align*}
\]
Sample task: steady hand path tracing

M. Li et al.

Background: endoscopic sinus surgery

Goal: robotically-assisted sinus surgery

- Difficulties with conventional approach
  - Complicated geometry
  - Safety-critical structures
  - Limited work space
  - Awkward tools

- Our approach
  - Cooperatively controlled “Steady hand” robot
  - Registered to CT models
  - “Virtual fixtures” automatically derived from models

Experiment Setup
Experimental setup

- Plastic Skull Phantom
  - Target path defined by embedded wire
  - Radioopaque fiducials implanted on skull for registration
- Computer model
  - Extracted from CT scan using standard software (Slicer)
- 3D tracking of tools, etc. using Northern Digital Optotrak®
- Co-register model, robot, and optical tracker using standard techniques

M. Li et al.

Virtual Fixture Online Implementation

Registered model → Constraint generation → Robot interface

Path

Tool tip guidance virtual fixture

min \[ W \cdot (J_{tip} \cdot \Delta q - \Delta P_{des}) \]^2
Subject to
\[ G \cdot \Delta q \geq g \]

Registered model

Path

Tool tip guidance virtual fixture

State
• Anatomy – triangulated surface models
  • Patient-specific model of nose & sinus derived from CT
  • High complexity: 182,000 triangles & 99,000 vertices

• Tool shaft -- cylinder

• The boundary constraint generation requires us to find close-point pairs between boundary surface model & tool shaft

M. Li et al.

• Problem: How can we generate the right constraints in real time???
Our solution: efficient search method using covariance tree representation of model

Covariance trees:
- Williams & Taylor, 1998; other authors
- Variation of k-d trees
- Basic idea:
  - Hierarchically split 3D model into sub-volumes
  - Realign coordinate system of each sub-volume to align with moments of inertia
- Produces bounding boxes that closely approximate boundaries & fast searches

M. Li et al.
Our solution: efficient search method using covariance tree representation of model

Covariance trees:
- Williams & Taylor, 1998; other authors
- Variation of k-d trees
- Basic idea:
  - Hierarchically split 3D model into sub-volumes
  - Realign coordinate system of each sub-volume to align with moments of inertia
- Produces bounding boxes that closely approximate boundaries & fast searches

M. Li et al.

One difference from ICP problem

One difference from ICP problem:
- Here we in principle need to identify all anatomy that can interfere with tool shaft
- Consequently modify search to find all triangle edges that are closer than some threshold to tool shaft
- Further modify to prune search to eliminate redundant constraints
Control Implementation

- Formulate constrained least squares problem
- Constraints & objective function include terms for desired tip motion, joint limits, boundary constraints

\[
\xi = \min_{\Delta q} \left[ \begin{bmatrix} W_{\text{tip}} & W_k \\ W_{\text{joint}} \end{bmatrix} \cdot \begin{bmatrix} J_{\text{tip}}(q) \\ J_k(q) \\ I \end{bmatrix} \Delta q - \begin{bmatrix} \Delta P_{\text{tip-des}} \\ 0 \\ 0 \end{bmatrix} \right]
\]

subject to

\[
\begin{bmatrix} H_{\text{tip}} \\ H_k \\ H_{\text{joint}} \end{bmatrix} \cdot \begin{bmatrix} J_{\text{tip}}(q) \\ J_k(q) \\ I \end{bmatrix} (\Delta q) \geq \begin{bmatrix} h_{\text{tip}} \\ h_k \\ h_{\text{joint}} \end{bmatrix}
\]

Control Implementation

- Tip frame \( \Delta P_{\text{tip}} = J_{\text{tip}}(q) \cdot \Delta q \)

\[
\left[ \Delta P_{\text{tip}} - \Delta P_{\text{tip-des}} \right] \geq \text{THD}
\]

\[
\min \xi_{\text{tip}} = \left\| W_{\text{tip}} \cdot (J_{\text{tip}}(q) \Delta q - \Delta P_{\text{tip-des}}) \right\|
\]

subject to \( H_{\text{tip-des}} J_{\text{tip}}(q) \Delta q \geq h_{\text{tip}} \)

- Boundary constraint \( \Delta P_k = J_k(q) \cdot \Delta q \)

\[
\left[ W_i \cdot \Delta P_k \right] \geq \frac{n_k \cdot (P_k + \Delta P_k - P_k)}{d}
\]

\[
\min \xi_k = \left\| W_k J_k(q) \Delta q \right\|
\]

subject to \( H_k J_k(q) \Delta q \geq h_k \)

- Joints limitation \( \Delta q \)

\[
q_{\text{min}} - q \leq \Delta q \leq q_{\text{max}} - q
\]

\[
\min \xi_{\text{joint}} = \left\| W_{\text{joint}} \Delta q \right\|
\]

subject to \( H_{\text{joint}} \Delta q \geq h_{\text{joint}} \)
Control implementation

- Solve problem numerically with standard methods (Lawson & Hanson, 1974)
- Performance:
  - 6 ms/iteration on 2GHz Pentium 4 PC
  - Typically 20 to 39 constraints

Results

The average time in each control loop for the boundary searching is ~6ms
Results: Robot vs Freehand

Freehand Error: 1.8 ± 1.1mm

Robot Error: 0.8 ± 0.4 mm

Approx 1.5:1 improvement in time!
Combine constraints

Single Frame

Multiple Frame

Translational part

Rotational part

Select one or more

Customized virtual fixtures

5 Basic Geometric Constraints
(Virtual fixture library)

Optimization

Stay on a point

Move along a line

Maintain a direction

Prevent plane penetrating

Rotate around a line

M. Li, A. Kapoor

Engineering Research Center for Computer Integrated Surgical Systems and Technology

Copyright © R. H. Taylor
Example: Suturing

The suturing task involves
- Select entry and exit points
- Align (Move & Orient) Needle
- Bite: Pass Needle
- Loop
- Knot

Suturing: Align Step

0. Move Along a Line

1. Stay at a point + Rotate about a line
Suturing: Align Step

2. Stay at a point + Rotate about a line

3. Puncture

4. Stay at a point + Rotate about a line

M. Li, A. Kapoor

Suturing: Bite Step

- Ideal trajectory is a circle with radius equal to needle radius.
- Needle plane is parallel to entry and exit points and surface normal.

M. Li, A. Kapoor
Suturing: Results

The average error (mm) in ideal and actual points as measured by OptoTrak®

Preliminary data collected from 4 users 5 trials each.

<table>
<thead>
<tr>
<th>Error</th>
<th>Entry (mm)</th>
<th>Exit (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robot</td>
<td>0.6375; ( \sigma = 0.12 )</td>
<td>0.7742; ( \sigma = 0.37 )</td>
</tr>
<tr>
<td>Manual</td>
<td>--</td>
<td>2.1; ( \sigma = 1.2 )</td>
</tr>
</tbody>
</table>

- Suturing task using VF showed significant improvement in performance over freehand.
- Can be performed at awkward angles
- Avoids multiple trials and large undesirable movements inside tissue.

M. Li, A. Kapoor
Hard and soft constraints

- Constraints on the task can be “hard” or “soft”
- The relative sizes depend on the procedure, ranging from micros to tenths of millimeter.
- Soft constraints allow the controller to accommodate uncertainties inherent in surgical procedures.

“Soft” constraint implementation

Suppose that we have a problem of the form

$$\Delta \tilde{a}_{\text{opt}} = \arg \min \| E(\Delta \tilde{q}) \|^2$$

subject to a constraint of the form

$$A_i(\Delta \tilde{q}) \leq b_i$$
“Soft” constraint implementation

But suppose we want to make the barrier “soft”. I.e., allow the robot to go beyond the barrier at increasing cost until it hits a harder barrier later.

Add an explicit slack $s_i$ and add a penalty term to the objective function

$$\Delta \mathbf{q}_{\text{des}} = \arg \min \ E(\Delta \mathbf{q})^2 + \eta_i s_i^2$$

subject to a constraint of the form

$$A_i(\Delta \mathbf{q}) - s_i \leq b_i$$

$$0 \leq s_i \leq s_{\text{up},i}$$

This process can be repeated several times to produce progressively steeper costs.

Example: Stay near a point

Target Position: $x_0$

After incremental motion

$$x_p + \Delta x_p$$ close to $x_0$

We want...

$$A(x, s) = \| \delta_p + \Delta x_p \|^2 - s \leq \epsilon_1$$

where

$$\delta_p = x_p - x_0$$
Using Linear Constrained Quadratic Optimization

Matrix representation

\[ A \cdot \Delta \vec{x} - s \leq b \]

Use Constrained Least Squares to solve

\[
\arg\min_{\Delta \vec{x}} \| \Delta \vec{x} - \Delta \vec{x}^d \|^2 \\
\text{s.t. } A \cdot \Delta \vec{x} - s \leq b
\]

Linear approximation for constraints

- \( n \times m \) increase
  - Polyhedron approaches the inscribed sphere
  - Linearized conditions are a better approximation
  - More constraints require more time to solve the optimization problem
- Symmetrical polyhedron
  - \( n \times m = 4 \times 4 \)
- Bounded polyhedron
  - \( n \times m = 3 \times 3 \)
Example Task

• Constraint 1: Tip to move along curve C
• Constraint 2: Origin of \{s\} to move along
• Objective: Handle to follow user input

Results for Example Task

"Hard", \( w_{b,j} = 1 \times 10^{-3} \)

"Soft", \( w_{b,j} = 1 \times 10^{-3} \)
Nonlinear Optimization

• One problem with linearized least squares is the proliferation of constraints to approximate the real constraints

• Consequently, it is worth considering alternatives that can handle more general formulas “directly”

\[
\Delta \mathbf{q}_{\text{des}} = \arg \min_{\Delta \mathbf{q}} C(\Delta \mathbf{x}, \Delta \mathbf{q}, \mathbf{s})
\]

subject to

\[
\Delta \mathbf{x} = J \Delta \mathbf{q}
\]

\[
A(\Delta \mathbf{x}, \Delta \mathbf{q}, \mathbf{s}) \leq \mathbf{b}
\]

Using Non-Linear Constrained Optimization

• Use Sequential Quadratic Program* method

• SQP solves the following problem iteratively

\[
\arg \min_{\Delta \mathbf{q}} \nabla C(\bar{\mathbf{x}}(\bar{\mathbf{q}} + \Delta \mathbf{q}^k), \bar{\mathbf{s}}^k, \bar{\mathbf{x}}^d)^t \mathbf{d} + \frac{1}{2} \mathbf{d}^t \mathbf{B}^k \mathbf{d}
\]

s.t.

\[
\nabla A_{\mathbf{k}}(\bar{\mathbf{x}}(\bar{\mathbf{q}} + \Delta \mathbf{q}^k), \bar{\mathbf{s}}^k)^t \mathbf{d} \leq \bar{\mathbf{b}}_{\mathbf{k}}
\]

• Start with a solution \([\Delta \mathbf{q}^k, s^k]^t\)

• Descent direction along with step size determine next solution \([\Delta \mathbf{q}^{k+1}, s^{k+1}]^t\)


A. Kapoor, *et al.*
Remarks: Non-Linear Constraints

• Current incremental motion can be used as starting guess for next motion

• Worst case number of constraints \( n \) times \( m \), \( n = \# \) variables, \( m = \# \) nonlinear constraints

• Analytical gradient increases speed

Linear v. Non-Linear Constraints

Accuracy

Time

Joint #3 is constrained

Tip trajectory

Non-Linear

Linear

# Hyperplanes used for approximation
Effect of increasing control-loop time

- Large error at sharp turning
- Small interval reduces error

Interval: 150ms

Interval: 40ms

Ming Li et al., IROS '05

Example: Two-handed virtual fixture for centering knot with visual feedback

Model  Plan  Action

Patient-specific Information
- Images, lab results, genetics, etc.

General information
- Anatomic atlases, statistics, rules

Ankur Kapoor
Scalable Robot for Dexterous Surgery in Small Spaces (aka Snake Like Robot)

Team: A. Kapoor, Kai Xu, Wei Wei, N. Simaan, P. Kazanzides, R. H. Taylor
Collaborator: P. Flint, MD

Snake Like Robot
System Architecture
**DaVinci Master Robot**

**High Level Constrained Control**

- Joint Positions ($q_m$)
  - $F_{actual} = k_i(q_m)$
- Set Points ($q^d_m$)
  - $F_d = k_i(q^d_m)$
- Joint Velocities
- Current Frame(s) Info.
- Geometric Constraints on Frame(s)
- Optimization Framework
  - $\arg \min_{\dot{q}_m} \left\| W (\dot{x}_m - k \tau_m) \right\|_2$
  - s.t. $H_1 \dot{x}_m \geq h_1$
  - $H_2 \dot{x}_m \geq h_2$
  - $\dot{x}_m = J_m \dot{q}_m$
- From Slave…

---

**Master Side High-Level Controller**

- **Objectives:**
  - Minimize error between desired motion and actual motion
  - Oppose motion that increases master-slave tracking error
  - Minimize the extraneous motion of the joints, and
  - Avoid large incremental joint motions that could occur near singularities
Master Side High-Level Controller

- Constraints:
  - General form: $H_m \Delta q_m \geq h_m$
  - Not allow motion outside joint range
  - Not allow motion that exceeds joint velocity limits
  - Additional constraints can be added from the VF Library

DaVinci Slave Robot
High Level Constrained Control

- Joint Positions $(q_s)$
- Set Points $(q^d_s)$
- Joint Velocities
- Current Frame(s) Info.
- Geometric Constraints on Frame(s)
- Optimization Framework
  \[
  \arg \min_{q_s} \| W (\dot{x}_s - k \tau_s) \|_2 \\
  \text{s.t.} \quad H \dot{x}_s \geq h \\
  \dot{x}_s = J(q_s) \]
- From Master... $F_{\text{master}}$
- To Master... $F_S$
Slave Side High-Level Controller

**Objectives:**
- Minimize error between desired motion and actual motion
- Minimize the extraneous motion of the joints, and
- Avoid large incremental joint motions that could occur near singularities

**Constraints:**
- Not allow motion outside joint range
- Not allow motion that exceeds joint velocity limits
- **Collision avoidance between slaves**
- More constraints can be added from the VF Library
μForce Scaling Cooperative Control

Cooperative Control
Velocity at the tool (V) is proportional to (α gain) the user’s input force at the handle (Fh)

\[ \dot{x} = \alpha F_h \]

μForce Scaling
Amplifies (ϒ gain) the human-imperceptible forces sensed at the tool tip (Ft) to handle interaction forces (Fh) by modulating robot velocity.

\[ x - \alpha (F_h - \gamma F_t), \quad \text{e.g., } \gamma = 500 \]

Kumar \ et \ al. (ICRA’00); Balicki \ et \ al. (MICCAI’10); Uneri \ et \ al., BioRob 2010

μForce Guided Cooperative Control

- User fights against ever increasing resistance
- Ensure safety tip force limits
- User interaction is limited at high-resistance regions
- Try to avoid those regions for later peeling
- User gets “stuck”, gives up, tries re-approach
- Ensure continuous user motion, even at the boundaries

Uneri \ et \ al., BioRob 2010
μForce Guided Cooperative Control

- Global Limiting
  - Task-specific tip force limit
  - User controlled limit distribution
- Continuous motion at the constraint boundaries
- Virtual spring construct to ensure stability

\[
\begin{align*}
\dot{x}_{\text{lim}} &= \dot{x} \left( \frac{f_{\text{lim}} - |f_t|}{f_{\text{spring}}} \right) \\
\dot{x}_{\text{min}} &= k_p \left( 1 - s \frac{|f_t|}{|f_h|} \right) f_h
\end{align*}
\]

Local Force Minimization
- Guiding user towards direction of minimum resistance
- Sensitivity variable allows user override
- Haptically intuitive response
- Avoids / postpones reaching limits

Uneri et al., BioRob 2010

Experimental Platform

Focusing on:
- Properties of the tissue we interact with
- The method of interaction, i.e. performance of our algorithms

Performed on:
- Inner shell membrane of raw eggs
- Surrogate tissue for epiretinal membrane peeling

Uneri et al., BioRob 2010
Experiment:
Tissue Force Characterization

• A corrected position allows us to observe tissue strain
• Controlled constant force application
  – Incremented by 1mN, with 10s delay, over a range of 1-10mN
• Characteristic curve obtained reveals a similar pattern to those seen in fibrous tissue tearing
  – Toe region: Safe
  – Linear region: Predictive
  – Failure region: Peeling

Uneri et al., BioRob 2010

 Experiment:
\( \mu \)Force Guided Cooperative Control

• Task: delaminate PVC strip with acrylic adhesive from a wax surface.
• Strip is peeled at an average of 45°
• User was guided away from the centerline in the direction of lowest resistance

Uneri et al., BioRob 2010
Experiment: 
μForce Guided Cooperative Control

• Goal: Remove a section of egg inner shell membrane
• Circular trajectory consistent with the results from the strip peeling experiment
• Magnify the perception of tip forces lateral to direction of desired motion
• Results in a peel pattern seen Capsulorhexis maneuver

Uneri et al., BioRob 2010

Information-enhanced robotic surgery

[Diagram showing information-enhanced robotic surgery components]

- augmented reality displays imaging
- safety barriers shared control “virtual fixtures”

SAW
Information-enhanced robotic surgery

- Augmented reality displays
- Imaging
- Tool motions
- Safety barriers
- Shared control
- "Virtual fixtures"

- Fast local compliance
- Law
- "Virtual fixtures"
Virtual Fixture “Hook” in DaVinci API

- Experimental interface not in any clinical or commercial product.
- Specification developed jointly by JHU and Intuitive to support research
- Prototyped at JHU by Tian Xia and Russ Taylor
- Current version implemented in DaVinci “S” model by Lawton Verner at ISI, with “hooks” in a proprietary ISI Application Program Interface
- Accessed through cisst/SAW libraries

Compliance virtual fixtures

\[ \mathbf{F} = [\mathbf{R}, \mathbf{\dot{p}}] \]

- \( \mathbf{F}_c = [\mathbf{R}_c, \mathbf{\dot{p}}_c] \) = position compliance frame
- \( \mathbf{k}^{(+)}, \mathbf{k}^{(-)} \) = position stiffness factors
- \( \mathbf{b}^{(+)}, \mathbf{b}^{(-)} \) = damping factors
- \( \mathbf{g}^{(+)}, \mathbf{g}^{(-)} \) = force bias terms

- \( \mathbf{R}_o \) = orientation compliance frame
- \( \mathbf{k}_o^{(+)}, \mathbf{k}_o^{(-)} \) = orientation stiffness factors
- \( \mathbf{t}^{(+)}, \mathbf{t}^{(-)} \) = torque bias terms

\( t \) = time remaining on timeout counter
Compliance virtual fixtures

if \( t > 0 \) then
begin
\[ t = t - 1 \]
\[ \mathbf{q} = \mathbf{F}_{c} \mathbf{p} - R_{c}^{-1} \{ \mathbf{p} - \mathbf{p}_{c} \} \]
\[ \mathbf{v} = R_{c} \mathbf{p} \]
\[ \mathbf{h} = \mathbf{0}, \mathbf{\psi} = \mathbf{0} \]
for \( i \in \{ x, y, z \} \) do
\[ \begin{cases} \text{if } \mathbf{q}_{i} < 0 \text{ then } \mathbf{h}_{i} = \mathbf{g}_{i}^{-1} + \mathbf{k}_{i}^{-1} \mathbf{\dot{q}}_{i} + \mathbf{b}_{i} \mathbf{\dot{v}}_{i} \text{ else } \mathbf{h}_{i} = \mathbf{g}_{i}^{-1} + \mathbf{k}_{i}^{-1} \mathbf{\dot{q}}_{i} + \mathbf{b}_{i} \mathbf{\dot{v}}_{i} \end{cases} \]
\[ \mathbf{f} = R_{c} \mathbf{h}; \text{ add } \mathbf{f} \text{ to the forces exerted on the master} \]
\[ \mathbf{\dot{\theta}} = \text{Rodrigues vector corresponding to } \Delta \mathbf{R} = R_{c} \mathbf{R} \]
for \( i \in \{ x, y, z \} \) do
\[ \begin{cases} \text{if } \mathbf{\dot{\theta}}_{i} < 0 \text{ then } \mathbf{\dot{\psi}}_{i} = \mathbf{\eta}_{i}^{-1} + \mathbf{\kappa}_{i}^{-1} \mathbf{\dot{\eta}}_{i} \text{ else } \mathbf{\dot{\psi}}_{i} = \mathbf{\eta}_{i}^{-1} + \mathbf{\kappa}_{i}^{-1} \mathbf{\dot{\eta}}_{i} \end{cases} \]
add \( R_{c} \mathbf{\dot{\psi}} \) to the torques exerted on the master
end

Surface following virtual fixture

**Goal:** Stay on a surface; bias force drawing toward the surface; spring force resisting penetration

\[ \mathbf{p}_{c} = \text{closest point on surface} \]
\[ R_{z} \mathbf{z} = \text{surface normal at } \mathbf{p}_{c} \]

\[ \mathbf{k}^{(-)} = [0, 0, -\text{stiffness}] \]
\[ \mathbf{g}^{(+)} = [0, 0, -\text{bias}] \]

**Others** = 0
Curve following virtual fixture

**Goal:** Stay on a surface; bias force drawing toward the surface; spring force resisting penetration; follow curve on surface

- \( \mathbf{p}_c \) = closest point on line on surface
- \( R_c \hat{z} \) = surface normal at \( \mathbf{p}_c \)
- \( R_c \hat{x} \) = line tangent at \( \mathbf{p}_c \)

\[
\mathbf{k}^{(-)} = [0, -\text{follow stiffness}, -\text{penetration stiffness}]
\]

\[
\mathbf{g}^{(+)} = [0, 0, -\text{bias}]
\]

(Note: may just set to 0)

Others = 0

Surface following virtual fixture

**Goal:** Stay on a surface; bias force drawing toward the surface; spring force resisting penetration; torque to align to surface normal

- \( \mathbf{p}_c \) = closest point on surface
- \( R_c \hat{z} \) = surface normal at \( \mathbf{p}_c \)

\[
\mathbf{k}^{(-)} = [0, 0, -\text{stiffness}]
\]

\[
\mathbf{g}^{(+)} = [0, 0, -\text{bias}]
\]

\[
\mathbf{k}_o^{(+)} = \mathbf{k}_o^{(-)} = [-\text{orient stiffness}, -\text{orient stiffness}, 0]
\]

Others = 0
Limitation and Extensions

- The specific abstraction just presented has some limitations. In particular, it separates the position and orientation compliance in a way that makes coupling of orientations and translations non-trivial.
- This can be gotten around to some extent by continually updating the virtual fixture compliance parameters.
- There are several obvious extensions that may be tried. For example, one can provide fuller matrices for virtual fixture force/torque generation. E.g.:

$$\begin{align*}
\text{Compute } & \mathbf{q}, \mathbf{v}, \mathbf{\ddot{q}}, \mathbf{\ddot{v}} \text{ from } \mathbf{F}_e \text{ and } \mathbf{R}_v, \text{where } (\mathbf{\ddot{q}} = d\mathbf{\dot{q}} / dt) \\
\text{Compute a region } i \text{ of local configuration space from } & \mathbf{\tilde{q}} \text{ and } \mathbf{\tilde{v}} \\
\mathbf{h}^i & = K_i \begin{bmatrix} \mathbf{q} \\ \mathbf{\dot{q}} \end{bmatrix} + B_i \begin{bmatrix} \mathbf{\ddot{q}} \\ \mathbf{\dot{v}} \end{bmatrix} + \mathbf{g}_i
\end{align*}$$