Medical Robots, Constrained Robot Motion Control, and "Virtual Fixtures"

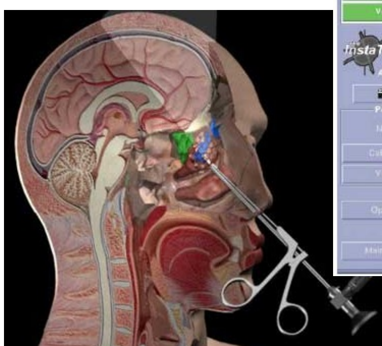
(Part 2)

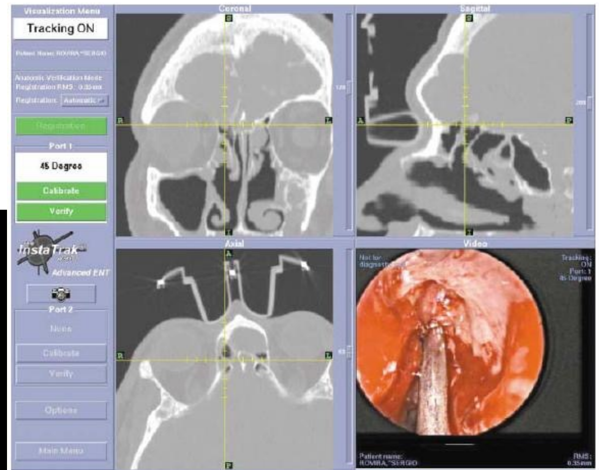
Russell H. Taylor
601.455/655

Disclosures & Acknowledgments

- **<u>This is the work of many people</u>**
- Some of the work reported in this presentation was supported by fellowship grants from Intuitive Surgical and Philips Research North America to Johns Hopkins graduate students and by equipment loans from Intuitive Surgical, Think Surgical, Philips, Kuka, and Carl Zeiss Meditec.
- Some of the work reported in this talk incorporates intellectual property that is owned by Johns Hopkins University and that has been or may be licensed to outside entities, including including Intuitive Surgical, Varian Medical Systems, Philips Nuclear Medicine, Galen Robotics and other corporate entities. Prof. Taylor has received or may receive some portion of the license fees. Also, Dr. Taylor is a paid consultant to and owns equity in Galen Robotics, Inc. These arrangements have been reviewed and approved by JHU in accordance with its conflict of interest policy.
- Much of this work has been funded by Government research grants, including NSF grants EEC9731478 and IIS0099770 and NIH grants R01-EB016703, R01-EB007969, R01-CA127144, R42-RR019159, and R21-EB0045457; by Industry Research Contracts, including from Think Surgical; by gifts to Johns Hopkins University from John C. Malone, Richard Swirnow and Paul Maritz; and by Johns Hopkins University internal funds.

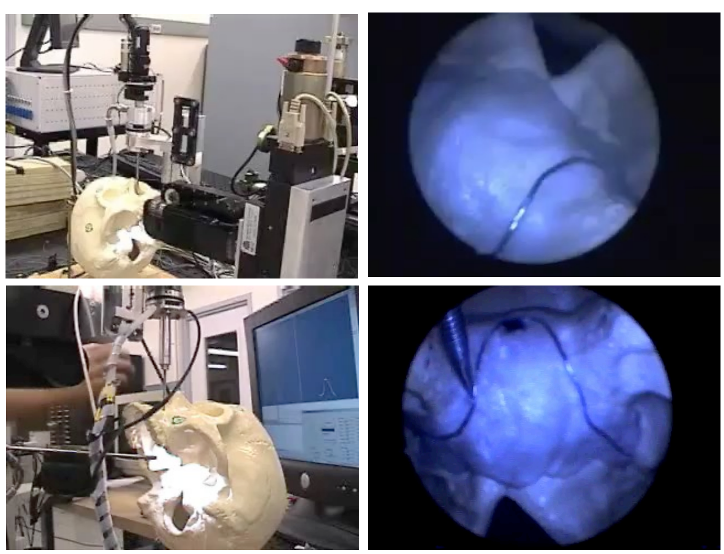
Typical application domain: endoscopic sinus surgery





Kennedy, D.W., W.E. Bolger , S. J. Zinreich , J. Zinreich, *Diseases of the Sinuses: Diagnosis and Management*. 2001.

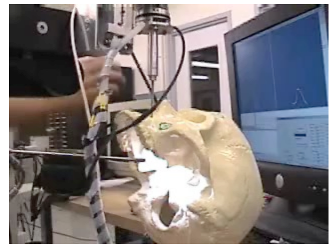
Sample task: steady hand path tracing

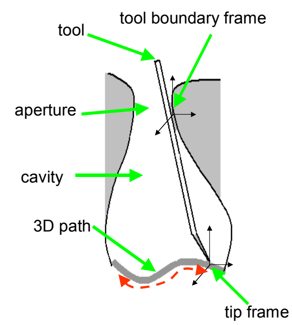


M. Li *et al.*

Goal: robotically-assisted sinus surgery

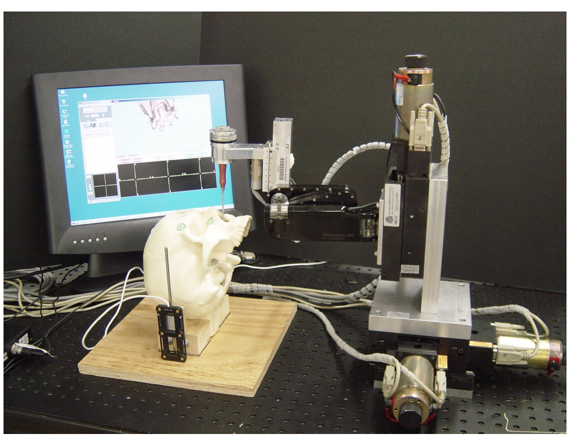
- **Difficulties with conventional approach**
  - Complicated geometry
  - Safety-critical structures
  - Limited work space
  - Awkward tools
- **Our approach**
  - Cooperatively controlled "Steady hand" robot
  - Registered to CT models
  - "Virtual fixtures" automatically derived from models

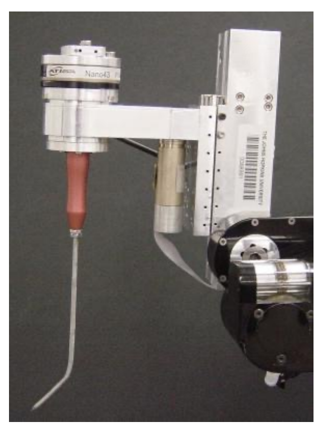




M. Li *et al.*

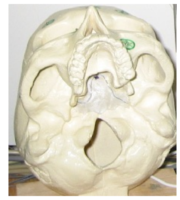
Experiment Setup



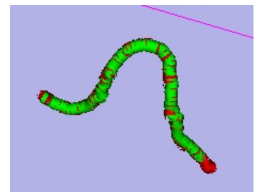


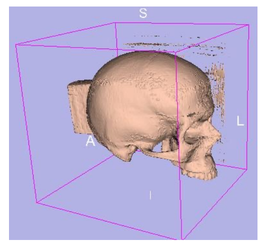
M. Li *et al.*

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



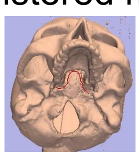


M. Li *et al.*

7

Virtual Fixture Online Implementation

Registered model



Path

Constraint generation

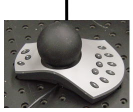
Tool tip guidance virtual fixture

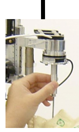
$$\min \left\| W \cdot \left(J_{tip} \cdot \Delta q - \Delta P_{des} \right) \right\|^2$$

Subject to

$$G \cdot \Delta q \geq g$$

Robot interface



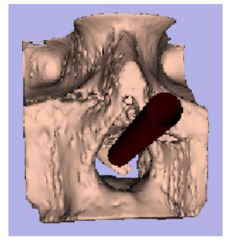


State

M. Li; R. Taylor; ICRA 2005

8

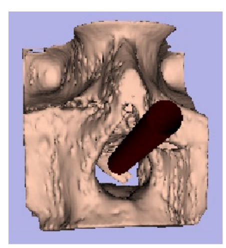
Boundary Constraints Generation



- 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.*

Boundary Constraints Generation

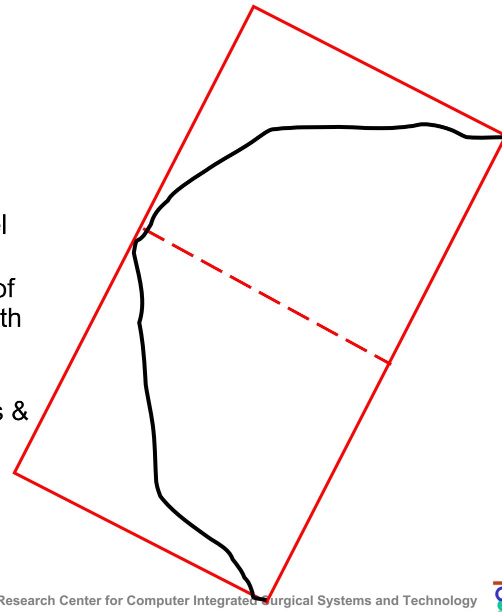


- Anatomy – triangulated surface models
  - Patient-specific model of nose & sinus derived from CT
  - High complexity: 182,000 triangles & 99,000 vertices
- Tool shaft -- cylinder
- The boundary constraint generation requires us to find close-point pairs between boundary surface model & tool shaft
- **Problem: How can we generate the right constraints in real time???**

M. Li *et al.*

Our solution: efficient search method using covariance tree representation of model

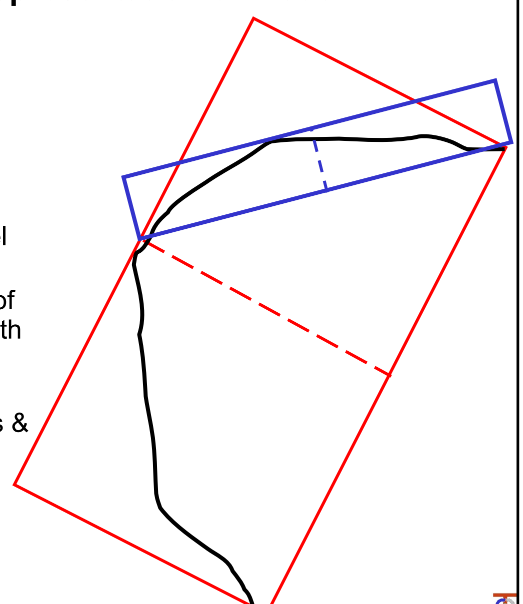
Covariance trees:
- Williams & Taylor, 1998; other authors
- Variation of k-d trees
- Basic idea:
  - Hierarchically split 3D model into sub-volumes
  - Realign coordinate system of each sub-volume to align with moments of inertia
- Produces bounding boxes that closely approximate boundaries & fast searches



M. Li *et al.*

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

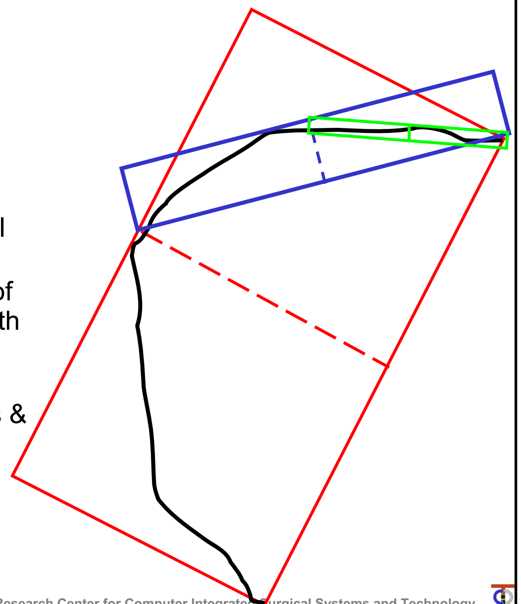


M. Li *et al.*

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

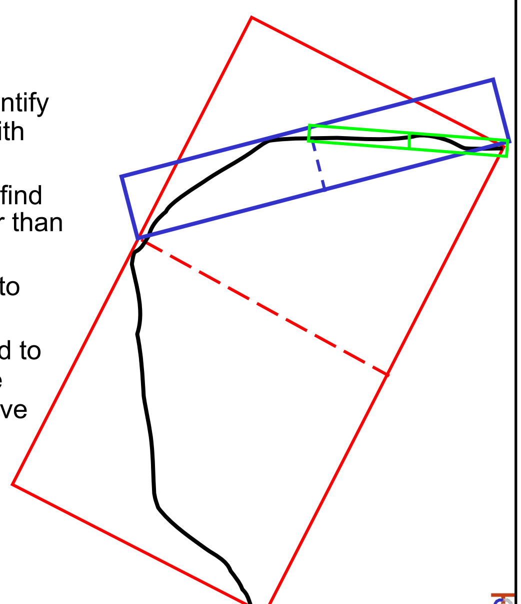


M. Li *et al.*

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
- **NOTE:** Generally, you only need to consider surfaces that are close enough so that the tool may move there in one time step.



Control Implementation

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

$$\zeta = \min_{\Delta q} \left\| \begin{bmatrix} W_{tip} & & \\ & W_k & \\ & & W_{joints} \end{bmatrix} \cdot \left(\begin{bmatrix} J_{tip}(q) \\ J_k(q) \\ I \end{bmatrix} \Delta q - \begin{bmatrix} \Delta P_{tip-des} \\ 0 \\ 0 \end{bmatrix} \right) \right\|$$

$$\text{subject to} \begin{bmatrix} H_{tip} & & \\ & H_k & \\ & & H_{joints} \end{bmatrix} \cdot \begin{bmatrix} J_{tip}(q) \\ J_k(q) \\ I \end{bmatrix} (\Delta q) \geq \begin{bmatrix} h_{tip} \\ h_k \\ h_{joints} \end{bmatrix}$$

M. Li *et al.*

Control Implementation

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

$\left\| \Delta P_{tip} - \Delta P_{tip-des} \right\|$

$\Delta P_{tip_d}^{T} \cdot \Delta P_{tip} \geq THD$

min $\zeta_{tip} = \left\| W_{tip} \cdot \left(J_{tip}(q) \Delta q - \Delta P_{tip-des} \right) \right\|$

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

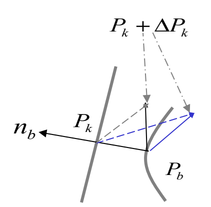
- Boundary constraint $\Delta P_k = J_k(q) \cdot \Delta q$

$\left\| W_k \cdot \Delta P_k \right\|$

$n_b^{T} \cdot (P_k + \Delta P_k - P_b) \geq d$

min $\zeta_k = \left\| W_k J_k(q) \Delta q \right\|$

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



- Joints limitation

$\left\| W_{joint} \cdot \Delta q \right\|$

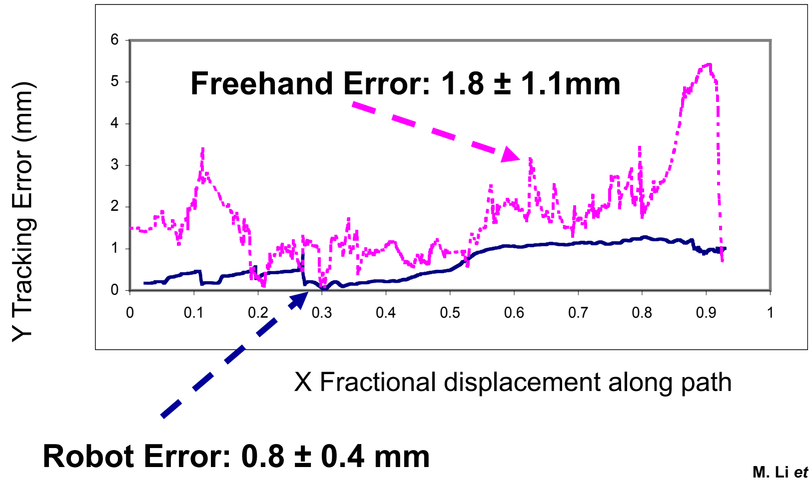
$q_{\min} - q \leq \Delta q \leq q_{\max} - q$

min $\zeta_{joints} = \left\| W_{joints} \Delta q \right\|$

subject to $H_{joints} \Delta q \geq h_{joints}$

M. Li *et al.*

Results: Robot *vs* Freehand



M. Li *et al.*

19

Results: Robot *vs* Freehand

Trial#	Free hand		Robot Guidance	
	Average Error (mm)	Average Time (s)	Average Error (mm)	Average Time (s)
1	1.785	26.354	0.736	18.972
2	1.632	29.358	0.757	15.275
3	1.796	27.372	0.765	16.29
4	2.061	25.436	0.779	19.439
5	2.119	24.533	0.777	16.209
avg	1.819	26.611	0.763	17.237
std	1.126	1.863	0.395	1.848

Approx 1.5:1 improvement in time!

M. Li *et al.*

20

Combine constraints

Single Frame

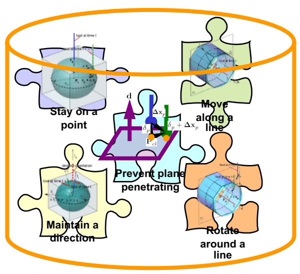
Translational part

$$\begin{bmatrix} A_p \\ A_r \end{bmatrix} J(q) \cdot \Delta q \le \begin{bmatrix} b_p \\ b_r \end{bmatrix}$$

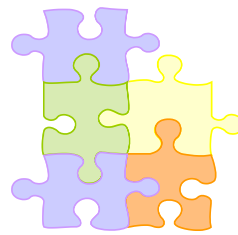
Rotational part

Multiple Frame

$$\begin{bmatrix} A_1, 0 \\ \ddots \\ 0, A_n \end{bmatrix} \begin{bmatrix} J_1(q) \\ \vdots \\ J_n(q) \end{bmatrix} \Delta q \le \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$



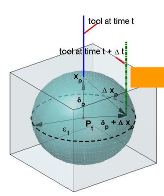
Select one or more



Customized virtual fixtures

M. Li, A. Kapoor

5 Basic Geometric Constraints
(Virtual fixture library)



Stay on a point

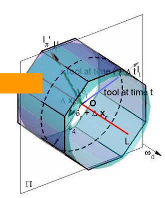
Optimization

$$\arg\min_{\Delta\vec{q}} C(\vec{x}(\vec{q}+\Delta\vec{q}), \vec{s}, \vec{x}^d)$$

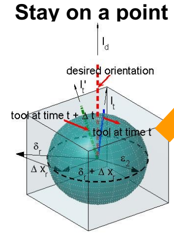
$$\text{s.t.} \quad A(\vec{x}(\vec{q}+\Delta\vec{q}), \vec{s}) \le \vec{b},$$

$$\vec{s}_{up} \ge s \ge \vec{s}_{low} \ge 0,$$

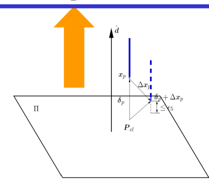
$$\Delta\vec{q}_{up} \ge \Delta\vec{q} \ge \Delta\vec{q}_{low}$$



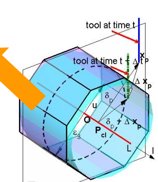
Move along a line



Maintain a direction



Prevent plane penetrating

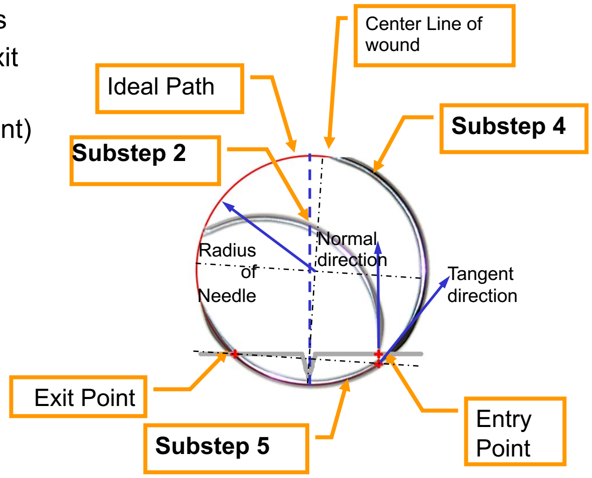


Rotate around a line

M. Li, A. Kapoor

Example: Suturing

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

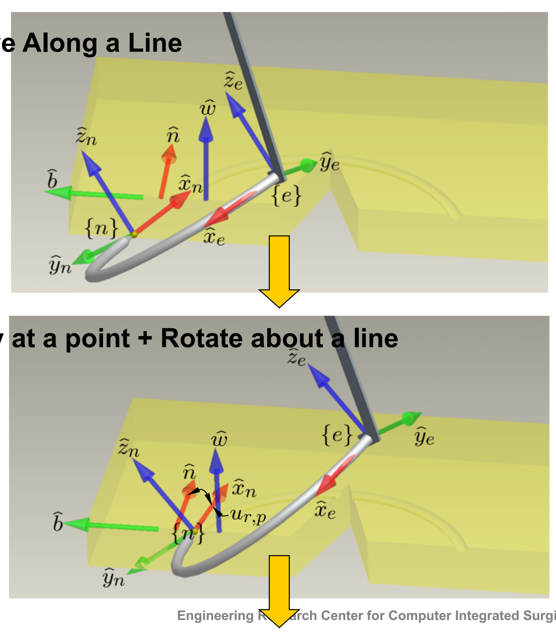


M. Li, A. Kapoor

Suturing: Align Step

0. Move Along a Line

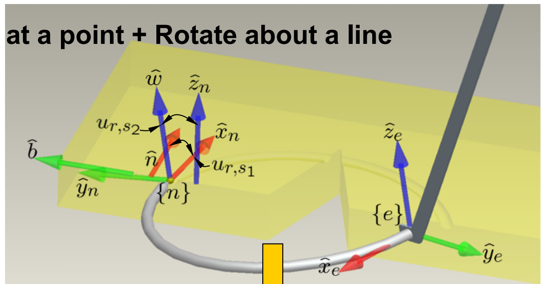
1. Stay at a point + Rotate about a line



M. Li, A. Kapoor

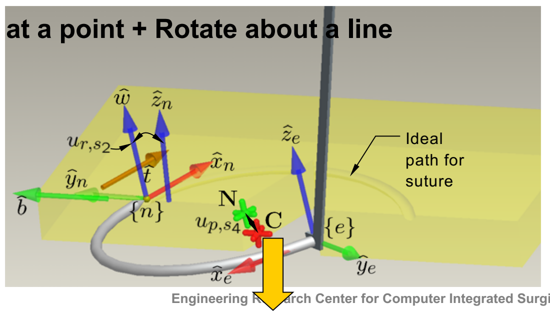
Suturing: Align Step

2. Stay at a point + Rotate about a line



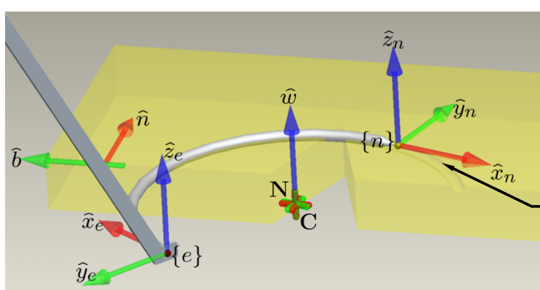
3. Puncture

4. Stay at a point + Rotate about a line



M. Li, A. Kapoor

Suturing: Bite Step

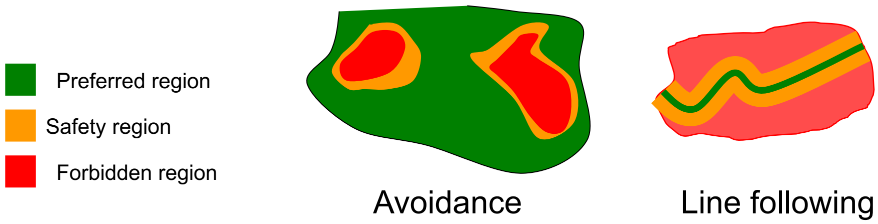


Ideal path for suture

- 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

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.

Thanks: **A. Kapoor**

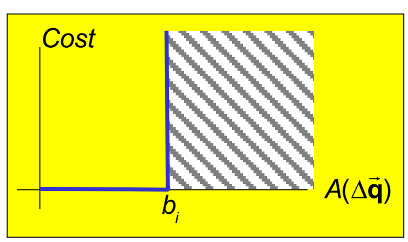
“Soft” constraint implementation

Suppose that we have a problem of the form

$$\Delta\vec{\mathbf{q}}_{des} = \arg\min \left\| E(\Delta\vec{\mathbf{q}}) \right\|^2$$

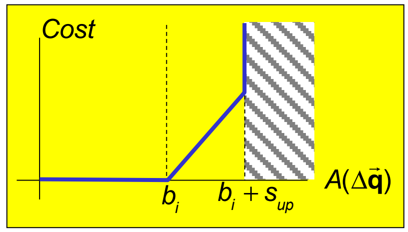
subject to a constraint of the form

$$A_i(\Delta\vec{\mathbf{q}}) \le b_i$$



"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

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

subject to a constraint of the form

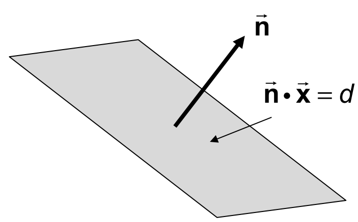
$$A_i(\Delta\vec{\mathbf{q}}) - s_i \leq b_i$$

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

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

Example

- Stay below a plane
- Move freely until get within 2 mm
- Increasing resistance as get close



$$\Delta\vec{\mathbf{q}}_{cmd} = \arg\min \ \eta_s s^2 + \sum_k \eta_k \left\|\cdots\text{other objective function terms}\right\|$$

such that

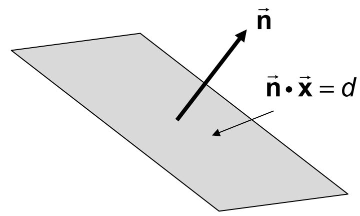
$$\vec{\mathbf{n}} \bullet (\vec{\mathbf{x}} + \Delta\vec{\mathbf{x}}) - s \leq d - 2$$

$$\Delta\vec{\mathbf{x}} = \mathbf{J}_{\vec{\mathbf{x}}}(\vec{\mathbf{q}})\Delta\vec{\mathbf{q}}$$

$$0 \leq s \leq 2$$

Example

- Stay below a plane
- Move freely until get within 2 mm
- Increasing resistance as get close



$$\Delta\vec{\mathbf{q}}_{cmd} = \arg\min \ \eta_s s^2 + \sum_k \eta_k \left\| \cdots \text{other objective function terms} \right\|$$

such that

$$\vec{\mathbf{n}} \bullet \Delta\vec{\mathbf{x}} - s \le d - 2 - \vec{\mathbf{n}} \bullet \vec{\mathbf{x}}$$
$$\Delta\vec{\mathbf{x}} = \mathbf{J}_{\vec{\mathbf{x}}}(\vec{\mathbf{q}})\Delta\vec{\mathbf{q}}$$
$$0 \le s \le 2$$

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"

$$\Delta\vec{\mathbf{q}}_{des} = \arg\min_{\Delta\vec{\mathbf{q}}} C(\Delta\vec{\mathbf{x}}, \Delta\vec{\mathbf{q}}, \vec{\mathbf{s}})$$

subject to

$$\Delta\vec{\mathbf{x}} = \mathbf{J}\Delta\vec{\mathbf{q}}$$
$$\mathbf{A}(\Delta\vec{\mathbf{x}}, \Delta\vec{\mathbf{q}}, \vec{\mathbf{s}}) \le \vec{\mathbf{b}}$$

Example: Stay near a point

Target Position: $\vec{x}_0$

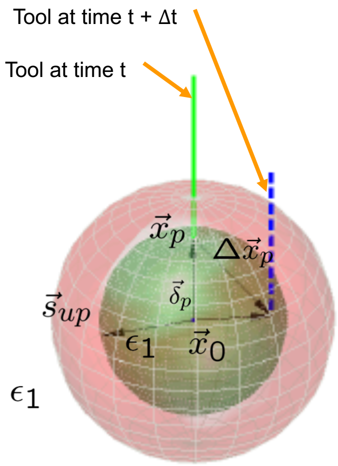
After incremental motion

$\vec{x}_p + \Delta\vec{x}_p$ close to $\vec{x}_0$

We want…

$$A(\vec{x}, s) = \|\vec{\delta}_p + \Delta\vec{x}_p\|^2 - s \le \epsilon_1$$

where $\vec{\delta}_p = \vec{x}_p - \vec{x}_0$



A. Kapoor, *et al.*

Using Linear Constrained Quadratic Optimization

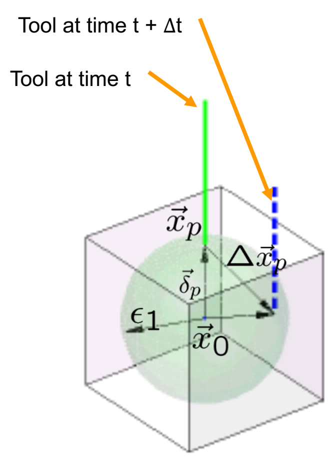
Matrix representation

$$A \cdot \Delta\vec{x} - s \le b$$

Use Constrained Least Squares to solve

$$\underset{\Delta\vec{q}}{\arg\min} \quad \|\Delta\vec{x} - \Delta\vec{x}^d\|^2$$

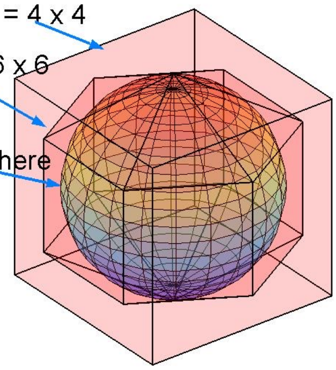
$$s.t \quad A \cdot \Delta\vec{x} - s \le b$$



A. Kapoor, *et al.*

Linear approximation for constraints

- n x 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
  - nxm = 4x4
- Bounded polyhedron
  - nxm = 3x3



A. Kapoor, *et al.*

Using Non-Linear Constrained Optimization

- Use Sequential Quadratic Program* method
- SQP solves the following problem iteratively

$$\mathbf{d}^{(k)} = \arg\min_{\mathbf{d}^{(k)}} \quad \nabla C\big(\mathbf{x}(\mathbf{q}+\Delta\mathbf{q}^{(k)}), \mathbf{s}^{(k)}, \mathbf{x}^d\big)^T \mathbf{d}^{(k)} + \frac{1}{2}\mathbf{d}^{(k)T}\mathbf{B}^{(k)}\mathbf{d}^{(k)}$$

$$\text{s.t.} \quad \nabla A_j\big(\mathbf{x}(\mathbf{q}+\Delta\mathbf{q}^{(k)}), \mathbf{s}^{(k)}\big)^T \mathbf{d}^{(k)} \le b_j; \quad j \in \mathcal{A}_k$$

- Start with a solution $[\Delta q^k, s^k]^t$
- Descent direction along with step size determine next solution $[\Delta q^{k+1}, s^{k+1}]^t$

*P. Spellucci, *Math. Prog.*, '98

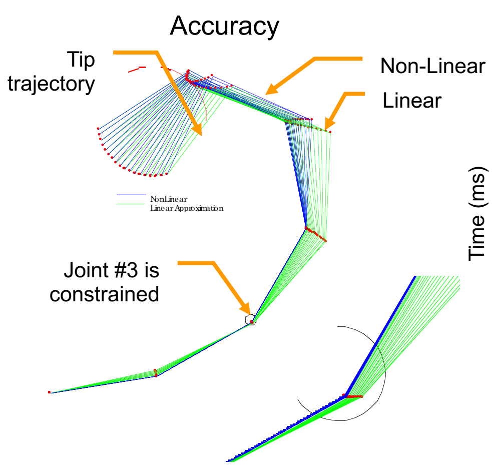
A. Kapoor, *et al.*

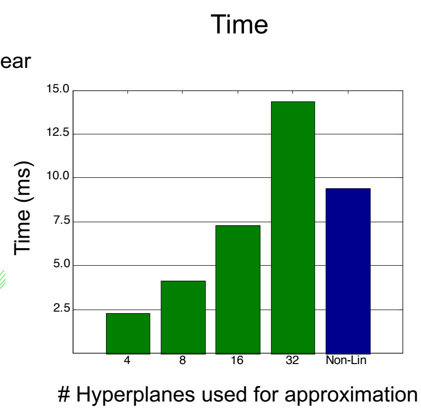
Remarks: Non-Linear Constraints

- 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

A. Kapoor, *et al.*

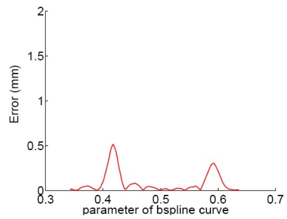
Linear v. Non-Linear Constraints



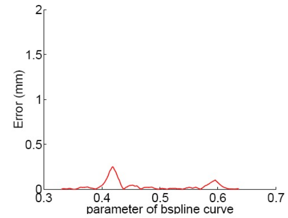


A. Kapoor, *et al.*

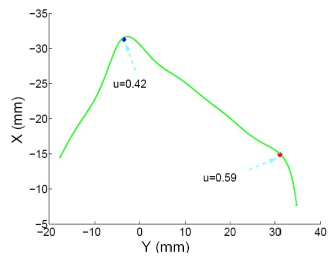
Effect of increasing control-loop time



Interval: 150ms



Interval: 40ms



- Large error at sharp turning
- Small interval reduces error

Ming Li et al., IROS '05

Longer Straight Line Motions

- In many cases, one wants to command a fairly long "straight line" motion from some initial pose to a final goal pose.
- This can be done fairly straightforwardly as follows:

$\mathbf{F}_0 =$ initial pose; $\mathbf{F}_G =$ goal pose;

$\dot{\theta}_{max} =$ max angular velocity; $v_{max} =$ max linear speed

Define $\mathbf{F}_{G0} = [\mathbf{R}_{G0}, \vec{\mathbf{p}}_{G0}]$ such that $\mathbf{F}_G \mathbf{F}_{G0} = \mathbf{F}_0$

Compute axis-angle representation for $\mathbf{R}_{G0} = Rot(\vec{\mathbf{n}}_{G0}, \theta_{G0})$

Compute $T_{move} = \max(\theta_{G0} / \dot{\theta}_{max}, \|\vec{\mathbf{p}}_{G0}\| / v_{max})$; $T_{left} = T_{move}$

while $T_{left} > 0$ do

  Wait for next time interval

  Perform housekeeping; input state ($\vec{\mathbf{q}}$, $\dot{\vec{\mathbf{q}}}$, forces, etc.)

  $T_{left} \leftarrow \max(T_{left} - \Delta T, 0); \lambda \leftarrow T_{left} / T_{max}$; $\mathbf{F}_T \leftarrow \mathbf{F}_G \bullet [\mathbf{R}(\vec{\mathbf{n}}_{G0}, \lambda\theta_{G0}), \lambda\vec{\mathbf{p}}_{G0}]$

  Set up optimization function to minimize $\left\| \mathbf{F}_T^{-1} \mathbf{F}(\vec{\mathbf{q}} + \Delta\vec{\mathbf{q}}) \right\|^2$

  Output velocity goal $\Delta\vec{\mathbf{q}} / \Delta T$

end

Longer Straight-Line Motions

To minimize $\left\| \mathbf{F}_T^{-1}\mathbf{F}(\vec{\mathbf{q}}+\Delta\vec{\mathbf{q}}) \right\|^2$ we actually want to try to make $\mathbf{F}_T \approx \mathbf{F}(\vec{\mathbf{q}}+\Delta\vec{\mathbf{q}}) = \mathbf{F}(\vec{\mathbf{q}})\Delta\mathbf{F}(\vec{\xi})$

in a least-squares sense, where $\vec{\xi} = \mathbf{J}_{kins}(\vec{\mathbf{q}})\Delta\vec{\mathbf{q}}$ and $\vec{\xi} = \left[\vec{\alpha}^T, \vec{\varepsilon}^T\right]^{T^{-1}}$

$$\Delta\mathbf{R} \approx \mathbf{R}(\vec{\mathbf{q}})^{-1}\mathbf{R}_T = Rot(\vec{\mathbf{n}}_{RT}, \vec{\theta}_{RT})$$
$$\vec{\mathbf{p}}_T \approx \mathbf{R}(\vec{\mathbf{q}})\vec{\varepsilon} + \mathbf{p}(\vec{\mathbf{q}})$$

This gives us the following minimization

$$\Delta\vec{\mathbf{q}} = \underset{\Delta\vec{\mathbf{q}}}{\operatorname{argmin}} \quad \nu_\alpha \left\| \vec{\alpha} - \vec{\theta}_{RT} \right\|^2 + \nu_\varepsilon \left\| \mathbf{R}(\vec{\mathbf{q}})\vec{\varepsilon} + \mathbf{p}(\vec{\mathbf{q}}) - \vec{\mathbf{p}}_T \right\|^2$$

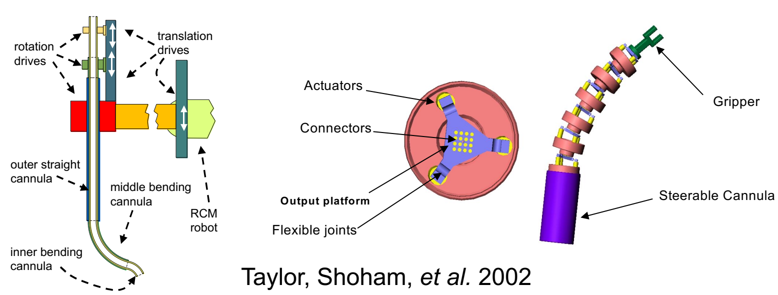
subject to

$\vec{\alpha} = \mathbf{J}_{kins}^{\alpha}(\vec{\mathbf{q}})\Delta\vec{\mathbf{q}} \quad \vec{\varepsilon} = \mathbf{J}_{kins}^{\varepsilon}(\vec{\mathbf{q}})\Delta\vec{\mathbf{q}}$

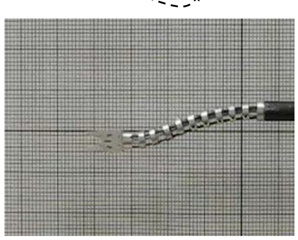
... other constraints such as joint limits, virtual fixture constraints, etc.

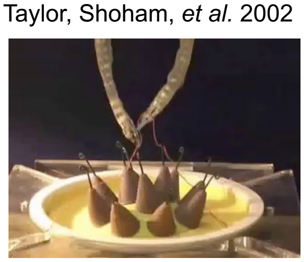
where ν_α and ν_ε can be used to control the relative importance of orientation and translation

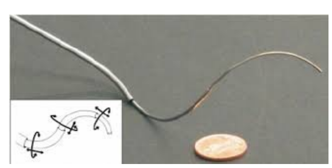
High Dexterity in a Small Package



Taylor, Shoham, *et al.* 2002



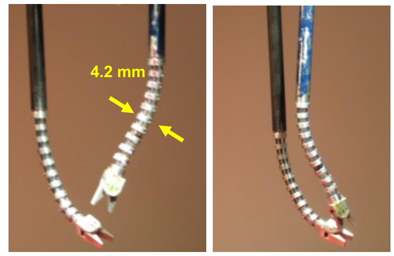


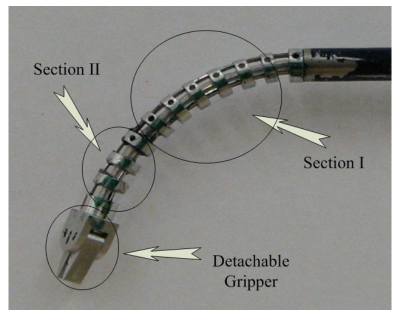


Webster *et al., 2006*

Simaan, Taylor *et al.* 2004, 2007

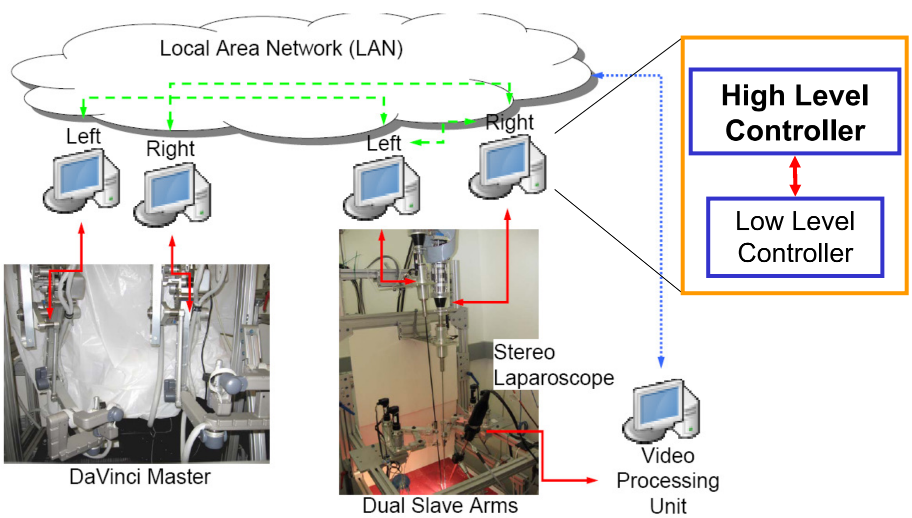
Scalable Robot for Dexterous Surgery in Small Spaces (aka Snake Like Robot)





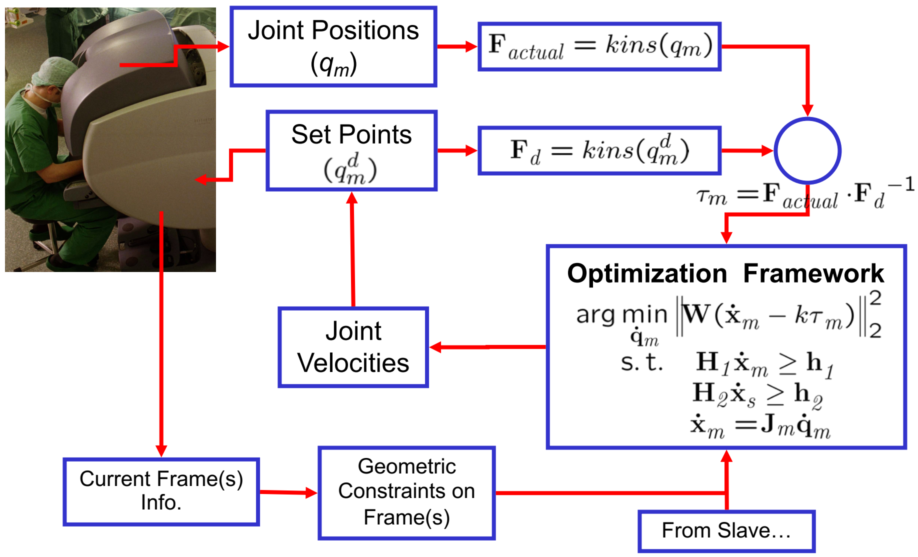
Team: A. Kapoor, Kai Xu, Wei Wei, N. Simaan, P. Kazanzides, R. H. Taylor
Collaborator: P. Flint, MD

Snake Like Robot
System Architecture

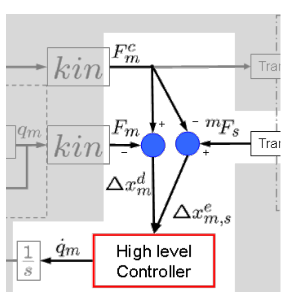


Master Tool Manipulator (MTM)

High Level Constrained Control



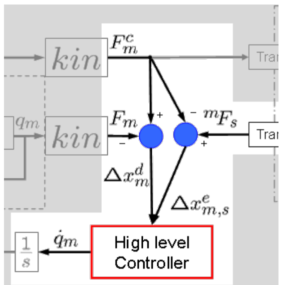
Master Side High-Level Controller



$$\min_{\Delta \vec{q}_m} \left\| W_{m,t} \left(\Delta x_m - K_a \left[p_m^e; \theta_m \omega_m \right] \right) \right\|$$
$$+ \left\| W_{m,s} \left(\Delta x_m - K_f \left[p_{m,s}^e; \theta_{m,s} \omega_{m,s} \right] \right) \right\|$$
$$+ \left\| W_{m,j} \Delta q_m \right\|$$

- **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



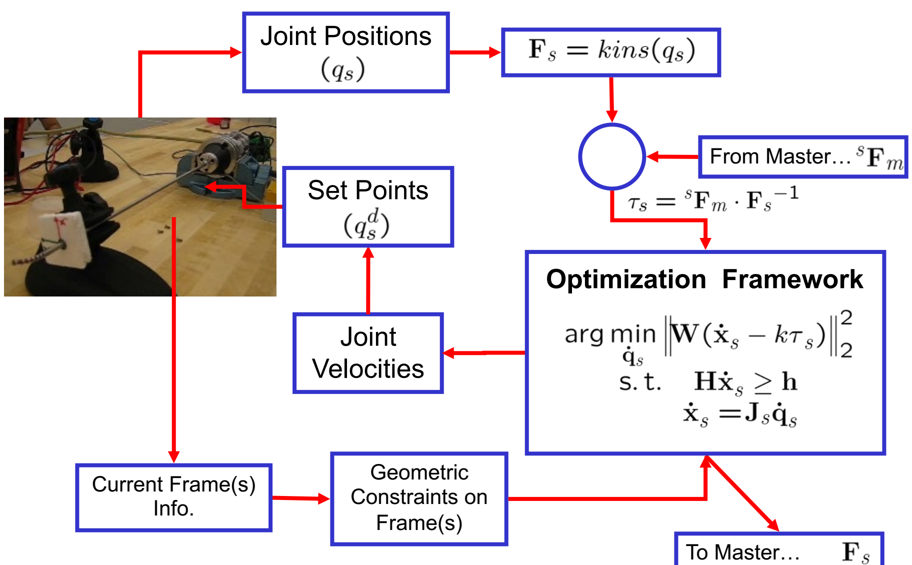
$$H_m \Delta q_m \geq h_m$$

that is $$\begin{bmatrix} I \\ -I \\ I \\ -I \end{bmatrix} \Delta q_m \geq \begin{bmatrix} q_{m,L} - q_m \\ q_m - q_{m,U} \\ \dot{q}_{m,U} \cdot \Delta t \\ \dot{q}_{m,U} \cdot \Delta t \end{bmatrix}$$

- **Constraints**:
  - General form: $H_{m,j}+\Delta q_m \geq h_{m,j}$
  - Not allow motion outside joint range
  - Not allow motion that exceeds joint velocity limits
  - Additional constraints can be added from the VF Library

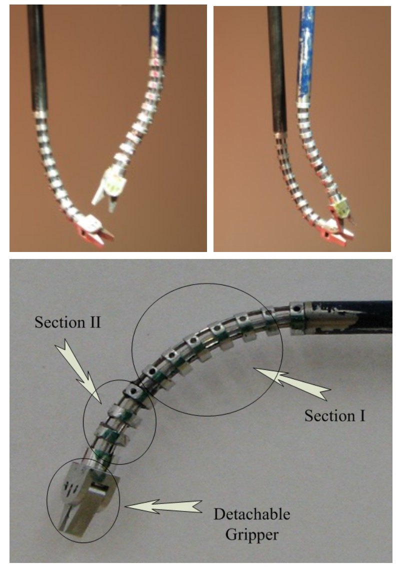
Patient-Side Manipulator (PSM)

High Level Constrained Control

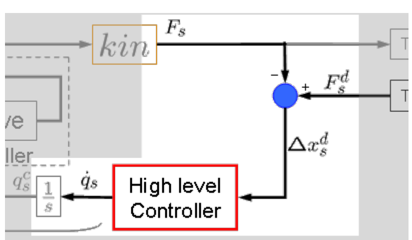


Patient-Side Snakes

- Actual snake section bends are a fairly complicated function of the linear displacements of the individual tubes and wires in the bending parts. But these displacements can be computed from the desired bending angles.
- Therefore, create pseudo-"joints" q_{sec1} and q_{sec2} corresponding to the bending angles in the two bend sections.
- Solve the optimization problem for q_{sec1} and q_{sec2} and the other joint angles of the slave robot. Then compute linear displacements from q_{sec1} and q_{sec2}. This also involves some calculations for redundancy resolution that can be done with a similar optimization method or can be done analytically.



Patient-Side High-Level Controller



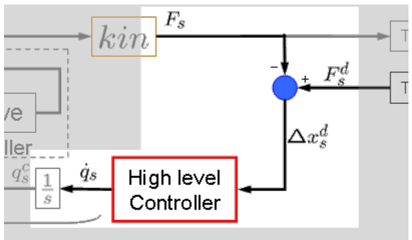
$$\min_{\Delta \vec{q}_s} \left\| W_{s,t} \left(\Delta x_s - K_a \left[p_s^e; \theta_s \omega_s \right] \right) \right\|$$

$$+ \left\| W_{s,j}(q) \Delta q_s \right\|$$

$$+ \left\| W_{s,} s \right\|$$

- **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

Patient-Side High-Level Controller



such that $\begin{bmatrix} I \\ -I \\ I \\ -I \end{bmatrix} \Delta q_s \geq \begin{bmatrix} q_{s,L} - q_s \\ q_s - q_{s,U} \\ \dot{q}_{s,U} \cdot \Delta t \\ \dot{q}_{s,U} \cdot \Delta t \end{bmatrix}$

and $\|\vec{d}\| + \Delta x_b \cdot \hat{d} + \vec{v} \cdot \hat{d} + s \geq d_{safe}$

$0 \leq s \leq s_{lim}$

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