Medical Robotics
Chapter II. Positioning

Guillaume.Morel@upmc.fr
Goals

• Acquire basic knowledge in 3D geometry for robotics:
  – Manipulation of homogeneous transforms, changing coordinate frames;
  – Kinematic modelling of robots;
  – Trajectories of bodies in space.

• Understand the difficulties associated with using a robot as a precise positioning device
  – Sources of uncertainty (sensor accuracy, deformations, backlash, control errors)
  – Calibration and registration procedures

• Understand how preoperative anatomical 3D models can be used for registration and / or navigation purposes.
Contents

1. Kinematic modeling
   – Parameterization of geometry and representation of configurations in SE(3)
   – Geometric modeling of robot manipulators (serial architectures)

2. Position control
   – Calculation of trajectories in space – interpolation
   – Position servoing

3. Principles of calibration and registration
   – Sensors
   – Algorithms
   – Accuracy and repeatability

4. Example of medical and surgical applications
   – Preoperative 3D anatomical models
   – Examples in navigation and robotics
   – In orthopedics: bone morphing; Casper
   – Protontherapy
   – Recalibration on deformable tissues: example of the prostate (Urostation).
1. KINEMATIC MODELLING
Direct and inverse kinematic models

Direct Kinematic Model

Joint configuration \( q \)

Inverse Kinematic Model

Operational configuration \( x \)
Example

How to:
• Define $q$ ?
• Define $x$ ?
• Compute $x = f(q)$ ?
• Compute $q = f^{-1}(x)$ ?
Joint parameters ($q$)

- A frame $F_i = (O_i, \hat{i}_i, \hat{j}_i)$ is attached to each body.
- An angle is defined between the axes of the successive marks.
- We regroup 3 angles in a vector of articual parameters: $q = [\theta_1 \ \theta_2 \ \theta_3]^T$. 
Operational parameters \((\mathbf{x})\)

\[
x = [x_T \ y_T \ \alpha]^T
\]
Direct kinematic model

\[ x = \begin{bmatrix} x_T \\ y_T \\ \alpha \end{bmatrix} = \begin{bmatrix} l_1 \cos \theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \\ l_1 \sin \theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \\ \theta_1 + \theta_2 + \theta_3 \end{bmatrix} \]
Inverse kinematic model

\[
\begin{bmatrix}
    x' \\
    y' \\
    \alpha
\end{bmatrix} =
\begin{bmatrix}
    x_T - l_3 \cos \alpha \\
    y_T - l_3 \sin \alpha \\
    \alpha
\end{bmatrix} =
\begin{bmatrix}
    l_1 \cos \theta_1 + l_2 \cos(\theta_1 + \theta_2) \\
    l_1 \sin \theta_1 + l_2 \sin(\theta_1 + \theta_2) + \\
    \theta_1 + \theta_2 + \theta_3
\end{bmatrix}
\]

\[
\begin{align*}
\theta_2 &= \arccos\left(\frac{x'^2 + y'^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \\
\theta_1 &= \text{atan2}(y', x') - \text{atan2}(l_1 \sin \theta_2, l_1 + l_2 \cos q_2) \\
\theta_3 &= \alpha - \theta_1 - \theta_2
\end{align*}
\]
General case

1.1. Operational parameterization
   – position
   – orientation
   – Homogeneous transforms

1.2. Joint parameterization
   – Denavit & Hartenberg convention
   – From measurement to DH parameters

1.3. Direct kinematic model

1.4. Inverse kinematic model
   – Pieper’s algorithm
   – Optimization formulation
1.1 Operational parameters

• Describing the geometric configuration of a frame with respect to another frame in SE(3) is equivalent to describing a displacement.

• 6 parameters are needed:
  – 3 to describe the change of origin
  – 3 to describe the change of orientation
Parameters for position (or translation)

- The 3 parameters for position are almost always the coordinates of the vector between the origin of the two frames:
  \[ t = O_0O_T \]
- These coordinates can be expressed in frame R₀:
  \[ p^0 = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \]
Parameters for orientation (or rotation)

- Not a vector space (not commutative).
- Not easy to manipulate, not easy to visualize.
- **In particular, one can not** parameterize orientation by “an angle $\alpha$ around $i$, an angle $\beta$ around $j$ and an angle $\gamma$ around $k$ ».
- Most often one uses three angles of elementary rotations around **SUCCESSIVE** angles (for interfaces and specifications).
- Internally, one uses rotation matrices for mathematical manipulation.

![Diagram of orientation parameters](image-url)
Rotation Matrices

\[
\begin{align*}
0_i_T &= (a_1 \ a_2 \ a_3)^T \\
0_j_T &= (b_1 \ b_2 \ b_3)^T \quad \Rightarrow \quad R_{0\rightarrow T} = \\
0_k_T &= (c_1 \ c_2 \ c_3)^T
\end{align*}
\]

- Only 3 independent parameters as:
  \[
  \begin{align*}
  i_T \cdot i_T &= j_T \cdot j_T = k_T \cdot k_T = 1 \\
  i_T \cdot j_T &= j_T \cdot k_T = k_T \cdot i_T = 0
  \end{align*}
  \]
- Rotational displacement: \[^1u' = R_{1\rightarrow 2} \cdot 1u\]
- Base change: \[^1u = R_{1\rightarrow 2} \cdot 2u\]
- Inverse rotation: \[R_{2\rightarrow 1} = (R_{1\rightarrow 2})^{-1} = (R_{1\rightarrow 2})^T\]
Composition of rotations

Through a right product:

\[ R_{1\to 3} = R_{1\to 2} R_{2\to 3} \]
Rotation Matrices for elementary rotations

- Rotation of an angle \( \alpha \) around vector \( i \): 
  \[
  R(\alpha, i) = \begin{pmatrix}
  1 & 0 & 0 \\
  0 & \cos \alpha & -\sin \alpha \\
  0 & \sin \alpha & \cos \alpha
  \end{pmatrix}
  \]

- Rotation of an angle \( \beta \) around vector \( j \): 
  \[
  R(\beta, j) = \begin{pmatrix}
  \cos \beta & 0 & \sin \beta \\
  0 & 1 & 0 \\
  -\sin \beta & 0 & \cos \beta
  \end{pmatrix}
  \]

- Rotation of an angle \( \gamma \) around vector \( k \): 
  \[
  R(\gamma, k) = \begin{pmatrix}
  \cos \gamma & -\sin \gamma & 0 \\
  \sin \gamma & \cos \gamma & 0 \\
  0 & 0 & 1
  \end{pmatrix}
  \]

- Rotation of an angle \( \theta \) around a unit vector \( u \):
  \[
  R(\theta, u) = \begin{pmatrix}
  u_x^2(1-c\theta) + c\theta & u_xu_y(1-c\theta) - u_zs\theta & u_xu_z(1-c\theta) + u_ys\theta \\
  u_xu_y(1-c\theta) + u_zs\theta & u_x^2(1-c\theta) + c\theta & u_yu_z(1-c\theta) - u_xs\theta \\
  u_xu_z(1-c\theta) - u_ys\theta & u_yu_z(1-c\theta) + u_xs\theta & u_x^2(1-c\theta) + c\theta
  \end{pmatrix}
  \]
  \[
  = (1-c\theta)uu^T + c\theta. I_{3x3} + s\theta. AS(u)
  \]
Euler angles (ZYX)

Euler angles define 3 successive rotations:

1. A rotation of an angle $\alpha$ around $k$ (Z axis)
2. A rotation of an angle $\beta$ around $j$ (Y axis) resulting from the first rotation.
3. A rotation of an angle $\gamma$ around $i$ (X axis) resulting from the two first rotations.
Euler angles to rotation matrices

- **Direct mapping:**
  \[
  \mathbf{R}_{0\to3} = \mathbf{R}_{0\to1}\mathbf{R}_{1\to2}\mathbf{R}_{2\to3} = \begin{pmatrix}
  \cos\alpha \cos\beta & \cos\alpha \sin\beta \sin\gamma - \sin\alpha \cos\gamma & \cos\alpha \sin\beta \cos\gamma + \sin\alpha \sin\gamma \\
  \sin\alpha \cos\beta & \sin\alpha \sin\beta \sin\gamma + \cos\alpha \cos\gamma & \sin\alpha \sin\beta \cos\gamma - \cos\alpha \sin\gamma \\
  -\sin\beta & \cos\beta \sin\gamma & \cos\beta \cos\gamma
  \end{pmatrix}
  \]

- **Inverse mapping**
  \[
  \mathbf{R} = \begin{pmatrix}
  r_{11} & r_{12} & r_{13} \\
  r_{21} & r_{22} & r_{23} \\
  r_{31} & r_{32} & r_{33}
  \end{pmatrix}
  \Rightarrow
  \begin{cases}
  \beta = \tan^{-1}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right) \\
  \alpha = \tan^{-1}\left(\frac{r_{21}}{\cos\beta}, \frac{r_{11}}{\cos\beta}\right), \text{ si } \cos\beta \neq 0 \\
  \gamma = \tan^{-1}\left(\frac{r_{32}}{\cos\beta}, \frac{r_{23}}{\cos\beta}\right), \text{ si } \cos\beta \neq 0
  \end{cases}
  \]

- **Singularity:**
  \[
  \begin{cases}
  \text{si } \beta = 90^\circ, \text{ alors } \alpha = 0 \text{ et } \gamma = \tan^{-1}\left(r_{12}, r_{22}\right) \\
  \text{si } \beta = -90^\circ, \text{ alors } \alpha = 0 \text{ et } \gamma = -\tan^{-1}\left(r_{12}, r_{22}\right)
  \end{cases}
  \]

(back)
Rotations around fixed axes: roll, pitch, yaw

Roll, pitch and yaw are defined by a succession of 3 rotations around fixed axes

1. A rotation of an angle $\alpha$ around vector $i_{\text{initial}}$ (X axis)
2. THEN a rotation of an angle $\beta$ around de vector $j_{\text{initial}}$ (Y axis)
3. THEN a rotation of an angle $\gamma$ around vector $k_{\text{initial}}$ (Z axis)

$$\mathbf{R}_{0\rightarrow 3} = \begin{pmatrix}
\cos\gamma \cos\beta & \cos\gamma \sin\beta \sin\alpha - \sin\gamma \cos\alpha & \cos\gamma \sin\beta \cos\alpha + \sin\gamma \sin\alpha \\
\sin\gamma \cos\beta & \sin\gamma \sin\beta \sin\alpha + \cos\gamma \cos\alpha & \sin\gamma \sin\beta \cos\alpha - \cos\gamma \sin\alpha \\
-\sin\beta & \cos\beta \sin\alpha & \cos\beta \cos\alpha
\end{pmatrix}$$
Quaternions

- Any rotation can be defined by: “turning by an angle \( \theta \) around a unit vector \( u \)”. Notice that the representation is not unique as \((-\theta, -u) \leftrightarrow (\theta, u)\)
- The vector \( \theta u \) is called finite rotation vector. From this vector, one can build quaternions.

\[
\begin{bmatrix}
\lambda_1 \\
\lambda_2 \\
\lambda_3 \\
\lambda_4
\end{bmatrix} = \begin{bmatrix}
\cos\left(\frac{\theta}{2}\right) \\
\sin\left(\frac{\theta}{2}\right)
\end{bmatrix} u
\]

\[
R = \begin{pmatrix}
2\lambda_1^2 + \lambda_2^2 & 2\lambda_2\lambda_3 - \lambda_1\lambda_4 & 2\lambda_2\lambda_4 + \lambda_1\lambda_3 \\
2\lambda_2\lambda_3 + \lambda_1\lambda_4 & 2\lambda_1^2 + \lambda_3^2 - 1 & 2\lambda_3\lambda_4 - \lambda_1\lambda_2 \\
2\lambda_2\lambda_4 - \lambda_1\lambda_3 & 2\lambda_3\lambda_4 + \lambda_1\lambda_2 & 2\lambda_1^2 + \lambda_4^2 - 1
\end{pmatrix}
\]

\[
R = \begin{pmatrix}
\frac{1}{2} \sqrt{r_{11} + r_{22} + r_{33} + 1} \\
\pm \frac{1}{2} \sgn(r_{32} - r_{23}) \sqrt{r_{11} - r_{22} - r_{33} + 1} \\
\pm \frac{1}{2} \sgn(r_{13} - r_{31}) \sqrt{r_{11} + r_{22} - r_{33} + 1} \\
\pm \frac{1}{2} \sgn(r_{21} - r_{12}) \sqrt{-r_{11} + r_{22} + r_{33} + 1}
\end{pmatrix}
\]

Homogeneous transforms

• Homogeneous coordinates of a point \( M \) in frame \( R_0 \) defined by:

\[
0_M = \begin{pmatrix}
    m_x \\
    m_y \\
    m_z \\
    1
\end{pmatrix}
\]

• Homogeneous coordinates of a vector \( v \) in the base of frame \( R_0 \) defined by:

\[
0_v = \begin{pmatrix}
    v_x \\
    v_y \\
    v_z \\
    0
\end{pmatrix}
\]

• Homogeneous transform between \( R_1 \) and \( R_2 \):

\[
T_{1 \rightarrow 2} = \begin{bmatrix}
    R_{1 \rightarrow 2} & 1_{t_{1 \rightarrow 2}} \\
    0 & 0 & 0 & 1
\end{bmatrix}
\]
Properties of homogeneous transforms

- $T_{1 \rightarrow 2}$ homogeneously represents the geometric transform (displacement) of points and vectors: $^1M' = T_{1 \rightarrow 2}^1M$, $^1v' = T_{1 \rightarrow 2}^1v$

- $T_{1 \rightarrow 2}$ homogeneously represents the coordinate change of points and vectors: $^1v = T_{1 \rightarrow 2}^2v$, $^1M = T_{1 \rightarrow 2}^2M$

- Inverse transform: $T_{2 \rightarrow 1} = (T_{1 \rightarrow 2})^{-1} = \begin{bmatrix} R_{1 \rightarrow 2}^T & -R_{1 \rightarrow 2}^T l_{1 \rightarrow 2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$

- Composition: $T_{1 \rightarrow 3} = T_{1 \rightarrow 2} T_{2 \rightarrow 3}$

- Decomposition into one translation followed by one rotation:

$$T_{1 \rightarrow 2} = \begin{bmatrix} R_{1 \rightarrow 2} & l_{1 \rightarrow 2} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} I_{3 \times 3} & l_{1 \rightarrow 2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{1 \rightarrow 2} & 0_{3 \times 1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
1.2 Joint parameters

• We are considering serial robots composed of a series of 1 degree of freedom joints.

• Each joint realizes either:
  – A rotation around a fixed axis (revolute or pivot joint)
  – A translation around a fixed axis (prismatic joint)

• Defining joint parameters comes down to answer the following 2 questions. Given a robot body:
  – How to describe its geometry with a minimal number of parameters?
  – How to describe its location with respect to the bodies to which it is connected?
General approach

• Given a robot body:
  – How to describe its geometry with a minimal number of parameters?
  – How to describe its location with respect to the bodies to which it is connected?

• Approach:
  – Attach a frame to each body with a repetitive and well-described method
  – Define geometric transforms between 2 successive frames
Dennavit & Hartenberg convention
Particular cases

• Frames $R_0$ et $R_n$ are placed using specific rules (undetermined problem).

• When $\Delta_i$ is not uniquely defined, i.e. when:
  – Two successive axes intersect (the direction of $i_i$ is to be chosen arbitrarily).
  – Two successive axes are parallel
  – Two successive axes are equal

then arbitrary choices are made that preserve the general convention, i.e., between 2 successive frames:

a) 1 translation by a distance $a_i$ along $i_i$;
b) 1 rotation of an angle $\alpha_i$ around $i_i$;
c) 1 translation by a distance $d_{i+1}$ along $k_{i+1}$;
d) 1 rotation of an angle $\theta_{i+1}$ around $k_{i+1}$
In practice

- Determine the joint axes and choose the sense of $z_i$;
- Place all the frames $R_i$ for which $\Delta_i$ is uniquely defined;
- Place the other frames for $1 \leq i \leq n-1$;
- Finally place $R_0$ (resp. $R_n$) supposing that it is identical to $R_1$ (resp. $R_{n-1}$) when $\theta_1 = d_1 = 0$ (resp. $\theta_{n-1} = d_{n-1} = 0$)
- Check that at the end, one still has between 2 successive frames:
  a) 1 translation by a distance $a_i$ along $i_i$;
  b) 1 rotation of an angle $\alpha_i$ around $i_i$;
  c) 1 translation by a distance $d_{i+1}$ along $k_{i+1}$;
  d) 1 rotation of an angle $\theta_{i+1}$ around $k_{i+1}$
- Group the results in a table:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\alpha_i$</th>
<th>$a_i$</th>
<th>$\theta_{i+1}$</th>
<th>$d_{i+1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example: SCARA robot

\[
\begin{align*}
\text{i} & \quad \alpha_i & \quad a_i & \quad \theta_{i+1} & \quad d_{i+1} \\
0 & \quad 0 & \quad 0 & \quad \theta_1 & \quad 0 \\
1 & \quad 0 & \quad a_1 & \quad \theta_2 & \quad 0 \\
2 & \quad 0 & \quad a_2 & \quad 0 & \quad d_3
\end{align*}
\]
Example: RPR robot

- [https://www.youtube.com/watch?v=-gcRPl-zTzw](https://www.youtube.com/watch?v=-gcRPl-zTzw)
From physical measurements to DH parameters

• In practice, each robot joint is equipped with a sensor that provides a measurement ($\theta_{m,i}$ for R joints, $d_{m,i}$ for P joints) with its own resolution and its own zero.

• One has to calibrate the robot to be able to convert these measurement into DH parameters:
  - For prismatic joints: $d_i = k_i d_{m,i} + d_{0,i}$
  - For revolute joints: $\theta_i = k_i \theta_{m,i} + \theta_{0,i}$

where $k_i$ is the gain accounting for the sensor resolution and transmission ratio and $d_{0,i}$ (resp. $\theta_{0,i}$) is the offset, i.e. the value of the DH parameter $d_i$ (resp. $\theta_i$) when the sensor measurement $d_{m,i}$ (resp. $\theta_{m,i}$) is zero.
Example: SCARA robot

- In practice, the measurement of the second joint is made with an optical encoder (2000 counts per revolution) mounted on a motor, connected to the robot body through gears with a gear ratio $N=1:87.6$.
- According to DH, when $\theta_2 = 0$, the arm is fully extended (the three axes are coplanar).
- The counter associated with the optical encoder increases when $\theta_2$ decreases according to DH.
- The counter associated with the optical encoder is set to zero during the initialization sequence when the arm is at $\theta_2 = 120$ degrees according to DH.
- Therefore: $\theta_2 = k_2 \theta_{m,2} + \theta_{0,2}$ with:
  - $k_2 = -\frac{360}{2000 \times 87.6} = -0.00205$ deg./count.
  - $\theta_{0,2} = 120$ deg.
Example: SCARA robot

- Finally the DH table looks like:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\alpha_i$</th>
<th>$a_i$</th>
<th>$\theta_{i+1}$</th>
<th>$d_{i+1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$k_1 \theta_{m,1} + \theta_{0,1}$</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$a_1$</td>
<td>$k_2 \theta_{m,2} + \theta_{0,2}$</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>$a_2$</td>
<td>0</td>
<td>$k_3 \theta_{m,3} + d_{0,3}$</td>
</tr>
</tbody>
</table>

![Diagram of SCARA robot](image)
1.3 Direct kinematic model

• Computing the direct kinematic model consists in establishing the function:
  \[ x = f(q) \]

• Starts with determining what’s \( x \) (generally 3 translations + 3 angles) and what’s \( q \) (according to DH convention).

• Then, thanks to the use of homogeneous transforms, it becomes completely automatic.
Computing the direct kinematic model

1. Attach frames to the robot bodies while respecting DH convention (i.e. build the table).

2. For $i = 1..n$, compute:

   $$T_{i-1\rightarrow i} = \begin{pmatrix}
   \cos \theta_i & -\sin \theta_i & 0 & a_{i-1} \\
   \sin \theta_i \cos \alpha_{i-1} & \cos \theta_i \cos \alpha_{i-1} & -\sin \alpha_{i-1} & -\sin \alpha_{i-1} d_i \\
   \sin \theta_i \sin \alpha_{i-1} & \cos \theta_i \sin \alpha_{i-1} & \cos \alpha_{i-1} & \cos \alpha_{i-1} d_i \\
   0 & 0 & 0 & 1
   \end{pmatrix}$$

3. Parameter the tool position and orientation, i.e. define $T_{n\rightarrow \text{tool}}$ and $x = f(T_{0\rightarrow \text{tool}})$.

4. Compute the product:

   $$T_{0\rightarrow \text{tool}} = T_{0\rightarrow n} T_{n\rightarrow \text{tool}} = \left( \prod_{i=1}^{n} T_{i-1\rightarrow i} \right) T_{n\rightarrow \text{tool}}$$

5. Extract from $T_{0\rightarrow \text{tool}}$ the parameters constituting vector $x$. 
Example : SCARA robot

\[ T_{i-1 \rightarrow i} = \begin{pmatrix}
\cos \theta_i & -\sin \theta_i & 0 & a_{i-1} \\
\sin \theta_i \cos \alpha_{i-1} & \cos \theta_i \cos \alpha_{i-1} & -\sin \alpha_{i-1} & -\sin \alpha_{i-1} d_i \\
\sin \theta_i \sin \alpha_{i-1} & \cos \theta_i \sin \alpha_{i-1} & \cos \alpha_{i-1} & \cos \alpha_{i-1} d_i \\
0 & 0 & 0 & 1
\end{pmatrix} \]
1.4 Inverse kinematic model

• No general and systematic method
• In general, no unique solution (see example in the introduction).
• When a formal expression is sought, one usually proceed “manually”. When possible, decouple orientation from position (see the example in the introduction or Pieper algorithm in the appendix).
• One usually proceed to a numerical inversion.
Numerical inversion

\[ q_{sol} = \arg\min_q \left( \left( \text{dist}(x, f(q)) \right)^2 \right) \]

Example solution = Newton-Euler algorithm, whose skeleton is:

1. \( q_{sol} \leftarrow q_0 \) (initial best guess)
2. While \( \|x - f(q_{sol})\| > \varepsilon \) do:
   - \( \delta q \leftarrow \lambda J^{-1}(q). [x - f(q_{sol})] \)
   - \( q_{sol} \leftarrow q_{sol} + \delta q \)

where \( J(q) \) is the jacobian matrix of \( f(q) \) whose element \( (i, j) \) is defined by \( \frac{\partial f_i}{\partial q_j} \) and \( \lambda \) is a gain.

Note that the algorithm can be coded in such a way that the convergence is guaranteed (e.g. adapting \( \lambda \)).

Of course, only one local solution is found (but frequently, from one solution, one can compute all of them).
2. POSITION CONTROL
Two distinct functions

• To position the end effector of a robot at a desired location (or along a desired trajectory), one needs:
  – An algorithm (or a system) that controls the actuators to servo the actual robot position at its desired value.
  – An algorithm that generates the desired position (e.g. a trajectory between given initial and final locations).
2.1 Trajectory interpolation

2.1.1 In joint space

• Input data:
  – Initial joint position $q_i$
  – Final joint position $q_f$
  – Maximal joint velocity $\dot{q}_{\text{max}}$
  – Maximal joint acceleration $\ddot{q}_{\text{max}}$

• Objective: compute the fastest trajectory given the limits
For 1 joint

- Compute $\delta = q_f - q_i$
- If $|\delta|$ small enough then the fastest trajectory consists in accelerating maximally then decelerating maximally (triangle velocity profile)
- Else, the velocity is saturated to its max value, leading to a trapeze profile.

- Question: knowing $\delta$, $\dot{q}_{max}$ and $\ddot{q}_{max}$, compute $t_1$, $t_2$ and $t_3$
Trajectory parameters computation

Knowing $\delta$, $\dot{q}_{max}$ & $\ddot{q}_{max}$, compute $t_1$, $t_2$ & $t_3$

- If $|\delta| \leq \frac{\dot{q}_{max}^2}{\ddot{q}_{max}}$ (triangle): $t_1 = t_2 = \sqrt{\frac{|\delta|}{\ddot{q}_{max}}} = \frac{t_3}{2}$

- Else (trapeze):
  - $t_1 = \frac{\dot{q}_{max}}{\ddot{q}_{max}}$
  - $t_2 = t_1 + \frac{|\delta| - \frac{\dot{q}_{max}^2}{\ddot{q}_{max}}}{\dot{q}_{max}}$
  - $t_3 = t_2 + t_1$
Trajectory generation

- Compute the desired position at a given time $t$, knowing the initial time $t_0$ (here set to zero for simplicity) and the trajectory parameters: $q_i$, $q_f$, $\dot{q}_{\text{max}}$, $\ddot{q}_{\text{max}}$, $t_1$, $t_2$ & $t_3$

- $\delta \leftarrow q_f - q_i$

- If $t < t_1$, then

$$q \leftarrow q_i + \text{sgn}(\delta q) \ddot{q}_{\text{max}} \frac{t^2}{2}$$

- Else if $t < t_2$ then

$$q \leftarrow q_i + \text{sgn}(\delta q) \ddot{q}_{\text{max}} \frac{t_1^2}{2} + \text{sgn}(\delta q) \dot{q}_{\text{max}} (t - t_1)$$

- Else if $t < t_3$ then

$$q \leftarrow q_i + \text{sgn}(\delta q) \ddot{q}_{\text{max}} \frac{t_1^2}{2} + \text{sgn}(\delta q) \dot{q}_{\text{max}} (t - t_1) - \text{sgn}(\delta q) \ddot{q}_{\text{max}} \frac{(t - t_2)^2}{2}$$

- Else

$$q \leftarrow q_f$$
Multi-joint interpolation

- Compute the profile for each joint,
- Select for $t_1, (t_2 - t_1)$ and $(t_3 - t_2)$ the slowest joint.
- Re-compute velocities and accelerations for all joints from $t_1, t_2$ and $t_3$
2.2.2 Interpolating in operational space.

- Operational parameters: $x = [x_p, x_o] = f(q)$.
- General case: $\text{dim}(x) = 6$.
- Input: Starting point $x_i$, End point $x_f$.
- General principles:
  - separate orientation from position.
  - Interpolate geometrically (path without time)
  - Select a velocity and acceleration along the path.
- Interpolation = find $x(\kappa), \kappa \in [0,1]$ such that:
  - Limit conditions are respected, i.e. $x(0) = x_i$ and $x(1) = x_f$.
  - $x(\alpha)$ is continuous (a minima), or (twice) continuously differentiable.
Interpolation for positions

• Rather simple: $x_p$ usually consists in the coordinates of a point $M$ in the robot base frame, one then linearly interpolates, i.e.:
  
  $$x_p(\kappa) = \kappa x_p(1) + (1 - \kappa)x_p(0)$$

• The resulting path is a straight line for point $M$, independently from the orientation of the end-effector that “turns around $M$". 
Interpolation for orientations

- Much more complex than for positions
- Usually, $x_o$ contains 3 angles that parameterize the orientation of the end-effector w.r.t. to the robot base.
- One can not interpolate linearly: $x_o(\kappa) = \kappa x_o(1) + (1 - \kappa) x_o(0)$
- Example: ZYX Euler angle, consisting in:
  1. A rotation of an angle $\alpha$ around $k$ (Z axis)
  2. A rotation of an angle $\beta$ around $j$ (Y axis) resulting from the first rotation.
  3. A rotation of an angle $\gamma$ around $i$ (X axis) resulting from the two first rotations.
Example of a linear interpolation on ZYX Euler angles

If \( x_0(0) = [0 \ 85 \ 0]^T \) and \( x_0(1) = [170 \ 85 \ -170]^T \) (both in degrees), then, through linear interpolation, one obtains:
Interpolation on a geodesic path

• Between two orientations, a distance can be defined: a geodesic distance.
• The geodesic distance is defined by the angle measured around the rotation axis separating the two orientations.
• Following a geodesic path consists in rotating around a fixed axis.
• **It is the shortest path between two orientations.**
• When following a geodesic path around a fixed axis \( u \), the rotational velocity is, at any time, collinear to \( u \). This underlines the similarity between the straight line path for positions and geodesic path for orientations, from a kinematic point of view: in both cases, the shortest path is obtained when the instantaneous velocity vector has a constant direction during the movement (independent from the time evolution of the magnitude).
Interpolation on a geodesic path

• General principle:
  – To determine a geodesic path between two known orientations $x_o(0)$ and $x_o(1)$, it is first required to extract angle $\theta$ and unit vector $u$ between the two orientations.
  – Then the interpolation consists in rotating around $u$ (which is kept constant) with an angle varying from 0 to $\theta$.

• Example from ZYX Euler angles:
  1. Building rotation matrices $R_{0\rightarrow T_i}$ from $[\alpha_i \ \beta_i \ \gamma_i]^T$, $i = 1,2$.

\[
R_{0\rightarrow T_i} = \begin{pmatrix}
\cos \alpha_i \cos \beta_i & \cos \alpha_i \sin \beta_i \sin \gamma_i - \sin \alpha_i \cos \gamma_i & \cos \alpha_i \sin \beta_i \cos \gamma_i + \sin \alpha_i \sin \gamma_i \\
\sin \alpha_i \cos \beta_i & \sin \alpha_i \sin \beta_i \sin \gamma_i + \cos \alpha_i \cos \gamma_i & \sin \alpha_i \sin \beta_i \cos \gamma_i - \cos \alpha_i \sin \gamma_i \\
-\sin \beta_i & \cos \beta_i \sin \gamma_i & \cos \beta_i \cos \gamma_i
\end{pmatrix}
\]
Interpolation on a geodesic path

2. Compute the rotational displacement:

\[ R_{T_0 \rightarrow T_1} = \left( R_{0 \rightarrow T_0} \right)^T R_{0 \rightarrow T_1} \]

3. extract angle \( \theta \) and unit vector \( u \) between the two orientations:

\[ R_{T_0 \rightarrow T_1} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \Rightarrow \begin{align*}
\theta &= \arccos \left( \frac{1}{2} \sqrt{r_{11} + r_{22} + r_{33} + 1} \right) \\
u &= \frac{1}{2 \sin \frac{\theta}{2}} \begin{pmatrix}
\text{sgn} (r_{32} - r_{23}) \sqrt{r_{11} - r_{22} - r_{33} + 1} \\
\text{sgn} (r_{13} - r_{31}) \sqrt{-r_{11} + r_{22} - r_{33} + 1} \\
\text{sgn} (r_{21} - r_{12}) \sqrt{-r_{11} - r_{22} + r_{33} + 1}
\end{pmatrix}
\end{align*} \]

4. Interpolation the angle \( \theta \): \( \theta(\kappa) = (1 - \kappa)\theta \).

5. Compute the corresponding rotation matrix:

\[ R_{T_0 \rightarrow T_\kappa} = \begin{pmatrix} 2(\lambda_1^2 + \lambda_2^2) - 1 & 2(\lambda_2 \lambda_3 - \lambda_1 \lambda_4) & 2(\lambda_2 \lambda_4 + \lambda_1 \lambda_3) \\
2(\lambda_2 \lambda_3 + \lambda_1 \lambda_4) & 2(\lambda_1^2 + \lambda_3^2) - 1 & 2(\lambda_3 \lambda_4 - \lambda_1 \lambda_2) \\
2(\lambda_2 \lambda_4 - \lambda_1 \lambda_3) & 2(\lambda_3 \lambda_4 + \lambda_1 \lambda_2) & 2(\lambda_1^2 + \lambda_4^2) - 1 \end{pmatrix} \]

With:

\[ \lambda_1 = \cos \left( \frac{\theta(\kappa)}{2} \right); \quad \begin{pmatrix} \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix} = \sin \left( \frac{\theta(\kappa)}{2} \right) u \]
Interpolation on a geodesic path

6. Compute the rotation between the base frame and frame $\alpha$:

$$R_{0\rightarrow T_\kappa} = R_{0\rightarrow T_0} R_{T_0\rightarrow T_\kappa}$$

7. Extract angles $x_o(\kappa)$.

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \Rightarrow \begin{cases} \beta = \tan 2 \left( -r_{31}, \sqrt{r_{11}^2 + r_{21}^2} \right) \\ \alpha = \tan 2 \left( \frac{r_{21}}{\cos \beta}, \frac{r_{11}}{\cos \beta} \right), \text{ if } \cos \beta \neq 0 \\ \gamma = \tan 2 \left( \frac{r_{32}}{\cos \beta}, \frac{r_{23}}{\cos \beta} \right), \text{ if } \cos \beta \neq 0 \end{cases}$$
Numerical example

\[ x_o(0) = [0 \ 85 \ 0]^T \text{ and } \]
\[ x_o(1) = [170 \ 85 \ -170]^T \]
2.2 Position servoing

- In joint space, when the robot is equipped with velocity controllers embedded in the power amps.

- Compensator $C_1(s)$: a simple gain often suffices.
Joint space position servoing with inner torque control (current)

- Compensator $C_1(s) = \text{PID}$:
  \[
  \gamma_c = K_p (q_d - q) + K_d (\dot{q}_d - \dot{q}) + K_i \int_0^t (q_d - q) \, d\tau
  \]
- $K_p =$ stiffness, $K_d =$ viscosity, $K_i$: no mechanical analogue, the torque varies (e.g. increases) as long as the error is not zero (thus compensating for disturbances like friction or weight).
Operational space control: exploiting the inverse kinematics

Initial and final desired position and orientation: $x(0), x(1)$

- Interpolation of operational space traj.: $x(\kappa)$ and $\kappa(t)$
- Inverse kinematic model

$\dot{q}_d$

Joint level compensator

Joint level control

Robot

- The closed loop dynamic is tuned at the joint level (e.g. joint by joint).
- Very easy to implement (used in many industrial controllers)
- For advanced controllers, one can time-differentiate the desired positions in order to obtain desired velocities and accelerations.
3. PRINCIPLES OF CALIBRATION AND REGISTRATION
Why?

• So far, we have seen methods based on proprioceptive sensors (i.e. joint sensors).
• From these measurements, one can control the trajectories of the end-effector (tool) w.r.t. the robot base.
• Unfortunately, this is useless if the environment location w.r.t. the robot base is not known, which usually happens in a surgical procedure.
• Registering the task means locating the task w.r.t. the robot base.
• Note: a lot of what follows is inspired from the book “Medical Robotics” edited by J. Troccaz, Wiley.
3.1 Algorithm

- A positioning task is defined in a frame.
- This frame’s location is unknown in the robot base.
- One needs to compute the geometric transform between the two frames.
- Principle:
  - Denoting the unknown transform by $T_{1 \rightarrow 2}$ between $R_1$ and $R_2$,
  - One measures features $^1F$ and $^2F$ in both frames.
  - One uses a similarity measurement $S(F_a, F_b)$ between features.
  - One determines $T_{1 \rightarrow 2}$ thanks to an optimizing procedure:
    $$\hat{T}_{1 \rightarrow 2} = \arg\max_{T_{1 \rightarrow 2}} S\left( ^1F, T_{1 \rightarrow 2}^2F \right)$$
Features

• Pixels or voxels (from 2D or 3D images of the same scene recorded from two different locations of a sensor)
• Pont coordinates in a plane (2D) or in space (3D).
• Surfaces (directly from surface sensors or through the segmentation of 3D images)
• Straight lines
• Etc.
Similarity

• Distance between paired point.

\[ S = \frac{1}{\sum_{i=1}^{n} \left\| \frac{1}{i} M_i \right\| \left\| \frac{1}{i} M'_i \right\|^2} \]

\[ x_1 \]
\[ y_1 \]
\[ O_1 \]

\[ x_2 \]
\[ y_2 \]
\[ O_2 \]

\[ x'_1 \]
\[ y'_2 \]
\[ O'_2 \]
A closed form solution

1. Given a cloud of \( n \) points \( M_i, i \in \{1..n\} \) whose coordinates are known in two frames \( R_1 \) and \( R_2 \) (respectively \( ^1M_i \) and \( ^2M_i \)).

2. Both clouds are first centered around the origin of their cloud:
\[
^1M_i' = ^1M_i - \frac{1}{n} \sum_{i=1}^{n} ^1M_i \quad \text{et} \quad ^2M_i' = ^2M_i - \frac{1}{n} \sum_{i=1}^{n} ^2M_i
\]

3. One then computes the covariance matrix and proceeds to an SVD decomposition:
\[
C = \begin{bmatrix} ^1M_1' & \ldots & ^1M_n' \\ ^2M_1' & \ldots & ^2M_n' \end{bmatrix}^T = UDV^T
\]

4. One then extracts an estimate of the rotation matrix and one for the translation:
\[
\hat{R}_{1 \rightarrow 2} = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det[VU^T] \end{bmatrix} V^T
\]
\[
\hat{t}_{1 \rightarrow 2} = \frac{1}{n} \sum_{i=1}^{n} ^1M_i - \hat{R}_{1 \rightarrow 2} \left( \frac{1}{n} \sum_{i=1}^{n} ^2M_i \right)
\]
Similarity

• Distance between points and surface (without pairing)

1. Registration is to be made w.r.t. a surface $S$, known in frame $R_O$.

2. With a sensor (in frame $R_C$) one measures $n$ points $C x_i$, $i \in \{1..n\}$ belonging to surface $S$.

3. One computes distance $d_i$ between each point $C x_i$ and its projection on the surface $S$, whose position depends on an estimate of $T_{O \rightarrow C}$: $d_i = f(C x_i, \hat{T}_{O \rightarrow C}, S)$.

4. One minimizes the quadratic sum of the distances:

$$[T_{O \rightarrow C}]_{\text{optimal}} = \underset{\hat{T}_{O \rightarrow C}}{\text{argmin}} \sum_{i=1}^{n} f^2(C x_i, \hat{T}_{O \rightarrow C}, S)$$
Iconic methods

• The features directly come from images (no segmentation).

• Direct use of pixels intensity $I$.
  
  – Sum of squared differences:

  $$SSD = \frac{1}{N} \sum_{i=1}^{N} [I_1(X_i) - I_2(X_i)]^2$$

  – Normalized cross correlation:

  $$NCC = \frac{\sum_{i=1}^{N} [I_1(X_i) - \bar{I}_1][I_2(X_i) - \bar{I}_2]}{\sqrt{\sum_{i=1}^{N} [I_1(X_i) - \bar{I}_1]^2 \sum_{i=1}^{N} [I_2(X_i) - \bar{I}_2]^2}}$$
3.2 Sensors

• Mechanical:
  – A poly-articulated arm, not actuated, equipped with joint sensors and a kinematic model.
  – Used to palpate points and thus identify their coordinates in sensory arm base frame.

• Magnetic
  – An antenna emits a magnetic field while the sensor is a coil receiving the field out of which the induced current is measured.

• Optical
  – Cameras (often used as stereo pairs)
  – (Spherical) visible targets
  – Characteristic points extraction
  – Structured light (e.g. laser grid pattern)

• Any other 2D or 3D (medical) images
Mechanical sensors

- **Pros:**
  - Easy setup
  - Reasonable cost

- **Cons:**
  - Installation and utilization time (one point only can be measured at a time)
  - Cumbersomeness
Optical sensors

• Pros:
  – Can measure simultaneously several points
  – “real-time” response

• Cons:
  – Sensitive to occlusions
  – Markers to be installed
Optical sensor + palpation tool (stylus)

1. The passive markers mounted on the stylus reflect the infrared light emitted by the base;
2. Triangulation is used to calculate the 3D position of the passive markers;
3. It is thus possible to locate a frame rigidly linked to the stylus;
4. When the stylus is calibrated, it is possible to know the position of the stylus tip in the sensor frame.
Magnetic sensors

• Pros:
  – No optical occlusion
  – There are very small models (easy miniaturization and sterilization)

• disadvantages
  – Magnetic "occlusions" (perturbation of the field by metallic objects)
  – Wire on sensor
Medical imagers (Troccaz 2011)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IRM intervent.</td>
<td>3D</td>
<td>Magnétisme</td>
<td>Tissus mous</td>
<td>Pas de métal</td>
<td>Très rare</td>
<td>Pseu-do</td>
<td>Neuro-chir.</td>
</tr>
<tr>
<td>Scanner intervent.</td>
<td>3D</td>
<td>Rayons X</td>
<td>Os, cavités aériennes</td>
<td>Irra-diant</td>
<td>moyen</td>
<td>Pseu-do</td>
<td>Bio-psies</td>
</tr>
<tr>
<td>Radiographie</td>
<td>Projection plane ou biplane, 3D</td>
<td>Rayons X</td>
<td>Os, cavités aériennes, cavités liquidiennes avec pdt de contraste</td>
<td>Irra-diant</td>
<td>3D relativement rare, 2D très courant</td>
<td>2D oui</td>
<td>Ortho, traumat, radio interv., etc.</td>
</tr>
<tr>
<td>Echographie</td>
<td>1D, 2D, bi-plane, 3D, 4D</td>
<td>ultrasons</td>
<td>Tissus mous, cavités liquidiennes</td>
<td>Très opérateur-dépendant</td>
<td>2D très courant</td>
<td>oui</td>
<td>Très large</td>
</tr>
<tr>
<td>Echographie Doppler</td>
<td>1D, 2D, 3D</td>
<td>Effet doppler</td>
<td>Flux liquidiens</td>
<td>2D très courant</td>
<td>oui</td>
<td>Cardio.</td>
<td></td>
</tr>
<tr>
<td>Endoscopie</td>
<td>2D ou 3D</td>
<td>Lumière visible</td>
<td>tout</td>
<td>2D très courant</td>
<td>oui</td>
<td>Très large</td>
<td></td>
</tr>
<tr>
<td>Microscopie</td>
<td>2D ou 3D</td>
<td>Lumière visible</td>
<td>tout</td>
<td>Très spécifique</td>
<td></td>
<td>Neuro-chir.</td>
<td></td>
</tr>
</tbody>
</table>

Tableau 1.1. Principales modalités d’imagerie per-opératoire (type de capteur, dimensionnalité des informations fournies, principe physique de l’acquisition, structures anatomiques visualisables, limites ou inconvénients, disponibilité dans les blocs opératoires, capacité à fournir des informations en temps réel, applications cliniques de préférence)
Calibration

• Calibrating a localization system is very similar, from an algorithmic point of view, to registering two systems. Only the unknowns change, consisting in:
  – Intrinsic parameters, defining the relation between 3D positions and the sensory signal (pixel, voxel, Volt, etc.)
  – Extrinsic parameters, defining the geometric transform between a known frame linked to the outer shape of the sensor and the frame in which the sensed data is obtained.

• Intrinsic parameters are specific to the sensor, while extrinsic parameters depend on how the sensor is installed in the experimental system.

• These parameters have constant values during the experiment and are needed to compute the localization of the device.
Example: Standard pinhole camera mounted in a robot effector

• Projection model maps 3D position in camera frame into pixels

\[
\begin{bmatrix}
u \\
v
\end{bmatrix} = f \begin{bmatrix} k_u & s_{uv} & c_X \\ 0 & k_v & c_Z \\ 0 & 0 & 1
\end{bmatrix} \begin{bmatrix} c_X \\ c_Y \\ c_Z \\ 1
\end{bmatrix} + \begin{bmatrix} c_u \\ c_v \\ 1
\end{bmatrix}
\]

• An homogeneous transform is used to link the camera frame to the robot end-effector frame:

\[
\begin{bmatrix} n_X \\ n_Y \\ n_Z \\ 1
\end{bmatrix} = T_{n\rightarrow c} \begin{bmatrix} c_X \\ c_Y \\ c_Z \\ 1
\end{bmatrix}
\]
Example: Calibrating a stylus

- Principle: rotating around a fixed point $M$ and measuring the position and orientation of the stylus frame $(s)$ w.r.t. the sensor base frame $(b)$
- One has: $bM = R_{b\rightarrow s,i}^sM + t_{b\rightarrow s,i}$
- Or: $[I_3 \quad -R_{b\rightarrow s,i}] \begin{bmatrix} bM \\ sM \end{bmatrix} = t_{b\rightarrow s,i}$
- For $n$ measures:
  $\begin{bmatrix} I_3 & -R_{b\rightarrow s,1} \\ \vdots & \vdots \\ I_3 & -R_{b\rightarrow s,n} \end{bmatrix} \begin{bmatrix} bM \\ sM \end{bmatrix} = \begin{bmatrix} t_{b\rightarrow s,1} \\ \vdots \\ t_{b\rightarrow s,n} \end{bmatrix}$
- Finally:
  $\begin{bmatrix} bM \\ sM \end{bmatrix} = \left[I_3 \quad -R_{b\rightarrow s,1} \right]^+ \begin{bmatrix} t_{b\rightarrow s,1} \\ \vdots \\ t_{b\rightarrow s,n} \end{bmatrix}$
  where (here): $A^+ = (A^T A)^{-1} A^T$
Example: calibration of an ultrasound probe (intrinsic and extrinsic)

- A physical phantom with known geometry
- An optical tracker that localizes both the probe and the phantom
- Image measures
- Calibrating = maximizing similarity between 3D measures and 2D measures.
4. MEDICAL AND SURGICAL APPLICATIONS OF REGISTRATION AND ROBOTIC POSITIONING
Example 1: CTBOT

• Application: a robot has a place a needle to deliver a treatment under CT Scan imaging

• From a registration perspective, this is a simple case as only two frames have to be registered: robot (or needle) frame and image (or CT Scan frame)

Images from the manual procedure
Any metallic part of the robot, including electrical connections and cables, are placed away from the imaging plane. Only the registration device is radio opaque in the field of view.
Marking

- An object is fabricated in a radio transparent material, that incorporates radio opaque metallic bars (a).
- As its geometry is known, this allows to locating the robot from one 2D slice image only (b).
- Registering = minimizing the distance, in the 2D slice plane, from the measured intersection points (white dots centroids) to the theoretical intersection points (where the points should be given a transform from the image frame to the object frame)
Final precision

Fig. 6.16: Résultat de positionnement relatif. (a) La pointe de l’aiguille entre en contact avec la cible matérialisée par une aiguille de faible diamètre (1 mm). (b) Image TDM obtenue dans cette configuration.

- A proton beam is used to precisely shoot a brain tumor
- The beam shape can be precisely conformed to the tumor shape.
- General workflow:
  - 3D multimodal preop imaging
  - Tumor contouring (manual)
  - Planning phase to compute the beams geometry
  - Per-op registration / positioning of the patient in front of the fixed proton beam generator
Pre-op (1)

Golden grains are implanted
Pre-op (2) : IRM-CT SCAN fusion

Automatic 3D registration
Pre-op (3) : planning

1. Tumor (targets) and risky zones (obstacles) contouring on each 2D slice.
2. Selection of incident angles for proton beams
3. Computation of beam shapes
4. Synthesis of a *Digitally Reconstructed Radiography*
Per-op : registration

$T_{tumor_{current} \rightarrow tumor_{final}}$

$= \arg\min_T \left( \sum_{i=1}^{n} d^2(T) \right)$

Where $d(T)$ is the distance, in the image, between the grains in DRR and the grains in the real image.
Per-op : positioning
Problem: dose / time

• Because of calibration imprecisions, and experimental factors, the positioning requires 2 to 4 loop runs to converge towards a null positioning error

• X ray doses for positioning then becomes an issue.

• Also, in 2007, time was an issue as Xray images were not digitalized.
One more step: decreasing the dose.

Polaris Vicra working space

Contention mask with visible target

Principle:
1: first registration using dual imaging: XRays + Polaris (so as to register)
2: iterative positioning with corrective loops using Polaris only
3: final Xray check for legal issues
Exercise

\[ F_{polaris} \]

\[ F_{XRay} \]

\[ F_{target\_current} \]

\[ F_{tumor\_current} \]

\[ F_{endeffect\_current} \]

\[ F_{robotbase} \]

\[ F_{target\_final} \]

\[ F_{tumor\_final} \]

\[ F_{endeffect\_final} \]
Exercise

• Inputs:
  – Preop:
    • Where should the tumor be w.r.t. the proton beam when finally positioned: $T_{beam\rightarrow tumor_{final}}$
    • DRR = X-Ray image of the patient when correctly positioned.
    • Room calibration: $T_{robotbase\rightarrow beam}, T_{robotbase\rightarrow polaris}, T_{robotbase\rightarrow XRay}$
  – Perop:
    • For registration: XRay images from which, thanks to a registration and DRR, one gets: $T_{tumor_{current}\rightarrow tumor_{final}}$
    • In real time, for any further loops, Polaris localization of the target: $T_{polaris\rightarrow target_{current}}$

• Output: $T_{robotbase\rightarrow robotend\_effector_{final}}$ (the final robot end effector position and orientation)
Solution. Step 1 registration

• From the Xray/DRR registration one gets: $T_{\text{tumor}_\text{current} \rightarrow \text{tumor}_\text{final}}$

• Compute where is the tumor w.r.t. target:

• Compute where is the end-effector w.r.t. tumor:

• Compute where should be the final target w.r.t. Polaris:
Solution. Step 2 compute $T_{\text{robot base} \rightarrow \text{end effector final}}$

from $T_{\text{polaris} \rightarrow \text{target current}}$

\[
T_{\text{robot base} \rightarrow \text{end effector final}} =
\]
Solution. Step 3 next loop steps (required to to imprecisions) $T_{robotbase\rightarrow endeffector_{final}}$ from

$T_{polaris\rightarrow target_{current}}$ and $T_{polaris\rightarrow target_{final}}$

$T_{robotbase\rightarrow endeffector_{final}} = $
Practical application

Real time measurement

Position OK

Robot command sent

Manual Validation

Medical Robotics – 2 – Positioning

oct.-17
Example 3: iconic registration with a non rigid model (TIMC - Sté Koelis, M. Baumann’s thesis, Grenoble)

- Application: ultrasound guided prostate biopsy

![Ultrasound guided prostate biopsy](image)

2D TRUS

![2D TRUS Image](image)
Hierarchical approach

I. Global Rigid
- Model of endorectal probe kinematics
  - 3 DOF
- Coarsest
- Systematic Search

II. Local Rigid
- Translation and rotation
  - 6
- Coarsest
- Local/Powell-Brent

III. Deformation
- Linear elastic deformations
  - ~100k
- Coarse to fine
- Variational/Full multigrid

\[ \theta^* [I_1, I_2; \varphi] = \arg \min_{\theta \in \Theta} \mathcal{D}[I_1, I_2 \circ \varphi(\theta)] \]
\[ \mathcal{D}_{cc} = \frac{\text{Cov}(I_1, I_2)}{\sqrt{\text{Var}(I_1)\text{Var}(I_2)}} \]
Example result: 3D monomodal iconic - elastic
Final rendering
Example 4 (Vitrani et al. IROS, 2008)

- Simultaneously calibrating an US probe + locating the probe w.r.t. a robot manipulating an instrument
The localization problem

Usually, the problem is solved by the use of external localizers (optical, magnetic) which are rather popular in computed assisted surgery and navigation (see paper for refs).
Modeling assumptions

Instrument

US Image

US beam

US probe

Image processing

Geometrical modelling
Geometrical Model

• From pixel to meter ( \([k_x, k_y] = \text{pixel/m}\) )

\[
\mathbf{s} = \begin{bmatrix} k_x & 0 \\ 0 & k_y \end{bmatrix} \begin{bmatrix} x_M \\ y_M \end{bmatrix}
\]

• Intersection between a line and a plane

\[
\begin{align*}
\mathbf{PM} \cdot \mathbf{k}_p &= 0 \\
\mathbf{O}_6 \mathbf{M} &= l \mathbf{k}_I
\end{align*}
\]

• Expressed in the probe frame:

\[
\mathbf{s} = \begin{bmatrix} k_x & 0 \\ 0 & k_y \end{bmatrix} \begin{bmatrix} (\mathbf{P} \mathbf{P}_0) x & + & (\mathbf{R}_{\mathbf{P} \rightarrow \mathbf{O}} \mathbf{O}_6) x & + & l(\mathbf{R}_{\mathbf{P} \rightarrow \mathbf{O}} \mathbf{O}_6) y & + & l(\mathbf{R}_{\mathbf{P} \rightarrow \mathbf{O}} \mathbf{O}_6) z & + & 0 \mathbf{k}_I \end{bmatrix}
\]

\[
\Rightarrow \quad \mathbf{s} = f(\mathbf{q}, \mathbf{p}) \quad \text{with} \quad \mathbf{p} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ r_p_x \\ r_p_y \\ r_p_z \\ k_x \\ k_y \end{bmatrix}^T
\]

\[
\mathbf{q} = \text{robot joint configuration}
\]
Calibration procedure

Comparison between measured coordinates and coordinates computed from the geometrical model:

– For several instrument positions (i), measure:
  
  • Joint configuration \( q_i \)
  • Instrument position in the image \( s_{\text{mes},i} \)
  
  and group them in \( q \) and \( s_{\text{mes}} \).

b) Start with a best guess estimate for \( p \) (preop data)

c) Compute a variation \( dp \) for \( p \)

\[
dp = J_p^+ (s_{\text{mes}} - f(q, p)) = J_p^+ (s_{\text{mes}} - s_{\text{rec}})
\]

d) Model validation using measured configurations which were not used for identification

with \( p = \begin{bmatrix} x_0 & y_0 & z_0 & r_{P_x} & r_{P_y} & r_{P_z} & k_x & k_y \end{bmatrix}^T \)
Jacobian Matrix

... the paper provides the Jacobian Matix computation, which leads to:

\[
J_{mi} = K \left[ -l \begin{bmatrix} k_6 \end{bmatrix}_x - \begin{bmatrix} d_{O_0O_6} \end{bmatrix}_x + \left( 1 - \frac{k_P^T dp_{O_6}}{k_P^T k_6} \right) k_6 k_P^T \begin{bmatrix} d_{O_0O_6} \end{bmatrix}_x \right]^T \right]
\]
Experimental set-up

• Robot Stäubli TX40
• Acuson Cypress Ultrasound machine and Siemens V2C3 probe
Experimental Results

Intensity
Gain = 1
Depth. = 21.5cm

Identification

Validation

$e_{\text{max}}(x) = 1.50\, \text{mm}$
$e_{\text{max}}(y) = 1.88\, \text{mm}$
$e_{\text{rms}}(x) = 0.52\, \text{mm}$
$e_{\text{rms}}(y) = 0$, oct.-17
Influence of the US machine tuning

- Gain = 1
- Depth = 10.8cm
\[ e_{\text{max}}(x) = 1.48\text{mm} \]
\[ e_{\text{max}}(y) = 2.20\text{mm} \]
\[ e_{\text{rms}}(x) = 0.88\text{mm} \]
\[ e_{\text{rms}}(y) = 0.84\text{mm} \]

- Depth = 21.5cm
\[ e_{\text{max}}(x) = 1.68\text{mm} \]
\[ e_{\text{max}}(y) = 1.65\text{mm} \]
\[ e_{\text{rms}}(x) = 0.68\text{mm} \]
Comments

• Although drastically simplified, the *plane-line intersection model* is valid with a ~1mm precision (=the resolution of the sensor).
• Localization is performed *without any additional sensors*.
• Extensive further experiments and simulations showed that:
  – 6 configurations only lead to similar precision.
  – No local minima was experimentally observed.
• Precision is not significantly affected by the tuning of the machine (which changes the magnification factors and the blob appearance).
CONCLUSIVE COMMENTS
Comments

• Among the nice functions that a robot can provide is positioning. It features:
  – Precision
  – Capacity of immobility for a very long time
• For interventions on a patient, it is required to register the robot w.r.t. the patient anatomy
• This may be fastidious and sensitive to measurement, parameters or manipulation errors.
• Prefer when the registration is direct, i.e. the instrument tip measures its relative location w.r.t. the anatomy ➔ less chained errors.
APPENDIX. PIEPER’S ALGORITHM
Appendix A. Pieper’s algorithm

- Applies to manipulators with 6 R joints, the three last axes coinciding in a single point P to form a spherical wrist.
- In frame $R_3$, the coordinates of P are known:

$$3P = \begin{pmatrix} 3p_x \\ 3p_y \\ 3p_z \\ 1 \end{pmatrix} = \begin{pmatrix} a_3 \\ -d_4 \sin \alpha_3 \\ d_4 \cos \alpha_3 \\ 1 \end{pmatrix}$$

- In frame $R_2$:

$$2P = \begin{pmatrix} 2p_x \\ 2p_y \\ 2p_z \\ 1 \end{pmatrix} = \begin{pmatrix} a_3 \cos \theta_3 + d_4 \sin \alpha_3 \sin \theta_3 + a_2 \\ a_3 \cos \alpha_2 \sin \theta_3 - d_4 \cos \alpha_2 \sin \alpha_3 \cos \theta_3 + d_4 \sin \alpha_2 \cos \alpha_3 - d_3 \sin \alpha_2 \\ a_3 \sin \alpha_2 \sin \theta_3 - d_4 \sin \alpha_2 \sin \alpha_3 \cos \theta_3 + d_4 \cos \alpha_2 \cos \alpha_3 + d_3 \cos \alpha_2 \\ 1 \end{pmatrix}$$
Appendix A. Pieper’s algorithm

- In frame $R_1$:

$$
1 P = \begin{pmatrix}
1 p_x \\
1 p_y \\
1 p_z \\
1
\end{pmatrix} = T_{1\rightarrow 2} \cdot 2 P = \begin{pmatrix}
\begin{bmatrix}
1 f_1(\theta_2, \theta_3) + a_1 \\
\cos \alpha_1 f_2(\theta_2, \theta_3) - \sin \alpha_1 f_3(\theta_2, \theta_3) \\
\sin \alpha_1 f_2(\theta_2, \theta_3) + \cos \alpha_1 f_3(\theta_2, \theta_3) \\
1
\end{bmatrix}
\end{pmatrix},
\begin{align*}
\begin{aligned}
f_1(\theta_2, \theta_3) &= \cos \theta_2 \cdot 2 p_x(\theta_3) - \sin \theta_2 \cdot 2 p_y(\theta_3) \\
\cos \theta_1 f_1(\theta_2, \theta_3) + a_1 - \sin \theta_1 &- \cos \theta_1 \cos \alpha_1 f_2(\theta_2, \theta_3) - \sin \alpha_1 f_3(\theta_2, \theta_3) \\
\sin \theta_1 f_1(\theta_2, \theta_3) + a_1 - \cos \theta_1 &- \\
\sin \alpha_1 f_2(\theta_2, \theta_3) + \cos \alpha_1 f_3(\theta_2, \theta_3)
\end{aligned}
\end{align*}
avec
\begin{align*}
f_2(\theta_2, \theta_3) &= \sin \theta_2 \cdot 2 p_x(\theta_3) + \cos \theta_2 \cdot 2 p_y(\theta_3) \\
\sin \alpha_1 f_2(\theta_2, \theta_3) + \cos \alpha_1 f_3(\theta_2, \theta_3)
\end{align*}

- In frame $R_0$, since $\alpha_0 = a_0 = d_1 = 0$:

$$
0 P = \begin{pmatrix}
0 p_x \\
0 p_y \\
0 p_z \\
0
\end{pmatrix} = T_{0\rightarrow 1} \cdot 1 P = \begin{pmatrix}
\begin{bmatrix}
\cos \theta_1 f_1(\theta_2, \theta_3) + a_1 - \sin \theta_1 \cos \alpha_1 f_2(\theta_2, \theta_3) - \sin \alpha_1 f_3(\theta_2, \theta_3) \\
\sin \theta_1 f_1(\theta_2, \theta_3) + a_1 - \cos \theta_1 \cos \alpha_1 f_2(\theta_2, \theta_3) - \sin \alpha_1 f_3(\theta_2, \theta_3) \\
\sin \alpha_1 f_2(\theta_2, \theta_3) + \cos \alpha_1 f_3(\theta_2, \theta_3)
\end{bmatrix}
\end{pmatrix},
\begin{align*}
\begin{aligned}
\cos \theta_1 f_1(\theta_2, \theta_3) + a_1 - \sin \theta_1 &- \cos \theta_1 \cos \alpha_1 f_2(\theta_2, \theta_3) - \sin \alpha_1 f_3(\theta_2, \theta_3) \\
\sin \theta_1 f_1(\theta_2, \theta_3) + a_1 - \cos \theta_1 &- \\
\sin \alpha_1 f_2(\theta_2, \theta_3) + \cos \alpha_1 f_3(\theta_2, \theta_3)
\end{aligned}
\end{align*}
avec
\begin{align*}
\cos \theta_1 f_1(\theta_2, \theta_3) + a_1 - \sin \theta_1 &- \cos \theta_1 \cos \alpha_1 f_2(\theta_2, \theta_3) - \sin \alpha_1 f_3(\theta_2, \theta_3) \\
\sin \theta_1 f_1(\theta_2, \theta_3) + a_1 - \cos \theta_1 &- \\
\sin \alpha_1 f_2(\theta_2, \theta_3) + \cos \alpha_1 f_3(\theta_2, \theta_3)
\end{align*}
Appendix A. Pieper’s algorithm

• Elimination of $\theta_1$:

\[
\begin{align*}
\|0 \mathbf{P}\|^2 &= 2a_1 (k_1(\theta_3) \cos \theta_2) + k_2(\theta_3) \sin \theta_2) + k_3(\theta_3), \\
0 \mathbf{p}_z &= \sin \alpha_1 (k_1(\theta_3) \sin \theta_2 - k_2(\theta_3) \cos \theta_2) + k_4(\theta_3),
\end{align*}
\]

avec \[\begin{align*}
k_1(\theta_3) &= \frac{2}{\sqrt{a_1}} \mathbf{p}_x(\theta_3), \\
k_2(\theta_3) &= -\frac{2}{\sqrt{a_1}} \mathbf{p}_y(\theta_3),
\end{align*}\]
et \[\begin{align*}
k_3(\theta_3) &= \|2 \mathbf{P}(\theta_3)\|^2 + a_1^2 + d_2^2 + 2d_2 \cdot 2 \mathbf{p}_z(\theta_3), \\
k_4(\theta_3) &= 2 \mathbf{p}_z(\theta_3) \cos \alpha_1 + d_2 \sin \alpha_1.
\end{align*}\]

• Elimination of $\theta_2$:

• si $a_1 = 0$, alors : \[\|0 \mathbf{P}\|^2 = k_3(\theta_3) = \|2 \mathbf{P}(\theta_3)\|^2 + d_2^2 + 2d_2 \cdot 2 \mathbf{p}_z(\theta_3).\]

• si $\sin \alpha_1 = 0$, alors : \[0 \mathbf{p}_z = k_4(\theta_3) = 2 \mathbf{p}_z(\theta_3) \cos \alpha_1.\]

• cas général : \[\left(\frac{\|0 \mathbf{P}\|^2 - k_3(\theta_3)}{4a_1^2}\right)^2 + \left(\frac{0 \mathbf{p}_z - k_4(\theta_3)}{\sin^2(\alpha_1)}\right)^2 = k_1^2(\theta_3) + k_2^2(\theta_3).\]

This last equation allows to solve for $\theta_3$. 
Appendix A. Pieper’s algorithm

- Compute $\theta_2$ then $\theta_1$.
- Compute the rotation induced by $\theta_1$, $\theta_2$ et $\theta_3$, then the rotation $R_{3\rightarrow 6}$ that has to be realized by the wrist:

$$R_{3\rightarrow 6} = \underbrace{(R_{0\rightarrow 3})^T}_{\text{déduit de } \theta_1, \theta_2 \text{ et } \theta_3} \cdot \underbrace{(R_{0\rightarrow 6})}_{\text{donnée du problème}}$$

- Invert the wrist geometry by identifying $R_{3\rightarrow 6}$ (similarly to the computation presented for the representation of orientations for 3 successive rotations)