Line of Cut Generation

Felix Jonathan, Michael Norris

April 1, 2016

Using similar concepts as the scissor training, we can generate the prediction of the line of cut.

1 Setup

1. Scissor with EM sensor attached rigidly
2. Patient / Phantom with EM sensor attached rigidly
3. EM pointer tool for generating septum surface data

2 Input Data

1. scissor pose from EM data (T_S^W)
2. patient head pose from EM data (T_F^W)
3. generated septum surface normal ($\vec{n}_{ss}^F$) and plane equation of the septum ($ax + by + cz + d = 0$)
4. scissor training data ($P_{sp}^s, \vec{n}_{sp}^s$)
5. scissor blade length ($BL \in \mathbb{R}$)

3 PseudoCode list

This pseudocode is used for finding point and vector that exist from the intersection of two planes defined in the same coordinate system.

Function *FindIntersection (plane paramater1, plane parameter2)*

Data: Plane parameter1: (a_1, b_1, c_1, d_1) and Plane parameter2: (a_2, b_2, c_2, d_2)

Result: $P_{intersection}, \vec{v}_{intersection}$, Validity

Initialize $\vec{n}_1 = [a_1, b_1, c_1]^T$, $\vec{n}_2 = [a_2, b_2, c_2]^T$

if $||n_1 - n_2|| < \varepsilon$ **then**

Print warning: "The planes are parallel to each other, therefore there is no possible line of cut."

return None, None, Not Valid

end

else

$\vec{v}_{intersection} = \vec{n}_1 \times \vec{n}_2$

Initialize plane parameter3 as a plane with $\vec{n}_3 = \vec{v}_{intersection}$ and $d_3 = 0$.

Initialize matrix $X = [d_1, d_2, d_3]^T$

Initialize matrix:

$$M = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix}$$

Since $\vec{n}_1, \vec{n}_2$, and $\vec{n}_3$ are always linearly independent, M will have an inverse.

$P_{intersection} = M^{-1} * X$

return $P_{intersection}, \vec{v}_{intersection}, Valid = true$

end

Algorithm 1: Finding Line Intersection between 2 planes

This pseudocode is used for finding point projection to 3D line defined in the same coordinate system

Function *PointToLineProjection (point pointA, point anyPointInLineB, vector LineB)*

Data: point A $P_A = (x_a, y_a, z_a)$, anyPointInLineB $P_0 = (x_0, y_0, z_0)$, and vector of lineB: $\vec{b}$

Result: $P_{projection}$

return $P_{projection} = P_A = P_0 + \frac{(P_A - P_0) \circ \vec{b}}{\vec{b} \circ \vec{b}} * \vec{b}$

Algorithm 2: Finding point projection on 3D line

This pseudocode is used for finding the line end point based on initial line point, line magnitude, and line direction defined in the same coordinate system

1 **Function** *FindLineEndPoint (point pointA, realNumber lineMagnitude, vector lineB)*

Data: point A $P_A = (x_a, y_a, z_a)$, lineMagnitude $l \in \mathbb{R}$, and vector of lineB: $\vec{b}$

Result: $P_{endPoint}$

2 **return** $P_{endPoint} = P_A + \frac{\vec{b}}{||\vec{b}||} * lineMagnitude$

Algorithm 3: Finding point projection on 3D line

This pseudocode is used for finding the line of cut prediction based on scissor pinch point position, scissor plane normal, septum plane normal, scissor sensor pose and face sensor pose defined various coordinate system

Function *GenerateLineOfCut (point scissorPinchPoint, vector scissorNormal, plane septumPlaneParameters, transform scissorSensorPose, transform faceSensorPose, realNumber bladeLength)*

Data: scissorPinchPoint: P^S_{sp}, scissorNormal $\vec{n}^S_{sp}$, septumPlaneParameters: (a^f, b^f, c^f, d^f), scissorSensorPose T^W_S, faceSensorPose T^W_f, bladeLength Bl

Result: Scissor pinch point in septum plane P^f_{sp}, a pair of points defining the of line of cut P^f_{start}, P^f_{end}, boolean variable defining lineValidity *valid*

Calculate scissor pose relative to face coordinate system. $T^f_S = ChangeReference(T^W_S, (T^W_f)^{-1})$

Calculate scissorPinchPoint position and scissorNormal relative to face coordinate system. $P^f_{sp} = ChangeReference(P^S_{sp}, T^f_S)$, $\vec{n}^f_{sp} = ChangeReference(\vec{n}^S_{sp}, T^f_S)$.

Initialize scissor plane parameter: $scissorPlaneParameters = (a^f, b^f, c^f, d^f)$. $(a^f, b^f, c^f) = \vec{n}^f_{sp}$ and $d_f = -\vec{n}^f_{sp} * P^f_{sp}$

Initialize point that exist in both scissor and face plane ($P^f_{intersection}$), and line of cut vector ($\vec{v}^f_{cut}$).

Calculate point, vector, and validity of the line. $P^f_{intersection}, \vec{v}^f_{cut}, valid = FindIntersection(scissorPlaneParameters, septumPlaneParameters)$

if *valid == false* **then**

  return $P^f_{sp}, P^f_{sp}, P^f_{sp}, valid$

end

else

  Calculate the scissor initial cutting point: $P^f_{start} = PointToLineProjection(P^f_{sp}, P^f_{intersection}, \vec{v}^f_{cut})$.

  Calculate squaredLineMagnitude: $lineMagnitude^2 = Bl^2 - (P^f_{start} - P^f_{sp})^2$

  if $squaredLineMagnitude \leq 0$ **then**

    valid = false.

    return $P^f_{sp}, P^f_{sp}, P^f_{sp}, valid$

  end

  else

    Calculate LineMagnitude: $lineMagnitude = \sqrt{lineMagnitude^2}$

    Calculate the scissor end cutting point: $P^f_{end} = FindLineEndPoint(P^f_{start}, lineMagnitude, \vec{v}^f_{cut})$ **return** $P^f_{sp}, P^f_{start}, P^f_{end}, valid$

  end

end

Algorithm 4: Finding point projection on 3D line

4 Processing Data

1. Estimate septum surface plane parameter. (See septum surface generation documentation for more information)
2. Load current scissor parameter.
3. Collect scissor and septum pose sensor data.
4. Use function GenerateLineOfCut to generate line of cut data.