University of Chiba and University of Monterrey

DIT

PPD

Real-time Inverse Kinematics Computation and Self-Collision Avoidance for a Master-Slave Robot

UNIVERSITY OF CHIBA

NAMIKI AKIO

NAMIKI@FACULTY.CHIBA-U.JP

UNIVERSITY OF MONTERREY

OSMAR ALCALÁ

STUDENT ID: 353229

Presentation

1. Introduction
2. System Configuration
3. Collision avoidance control
4. Experiment
5. Conclusion

Presentation

1. Introduction
2. System Configuration
3. Collision avoidance control
4. Experiment
5. Conclusion

Research Objective

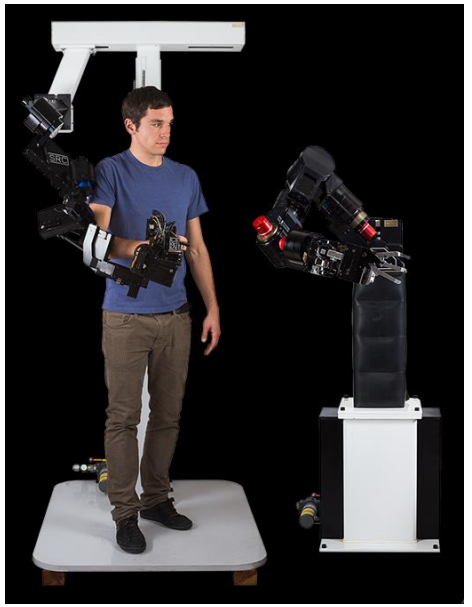
Human movement can be reckless some times so it is important that the robot does not collide against it’s own body trying to imitate this movements.

Unilateral Control → Unit Collision

Unilateral Control → Collision Avoidance

Background

Master-slave type robots are more suitable for dangerous environments than autonomous robots because they count with a human operator's judgement.



Designed for telerobotics

Wearing a special HMD and a pair of gloves, an operator controls a master-slave robot , Telesar V

Related Investigations

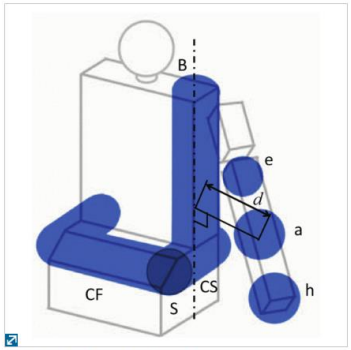
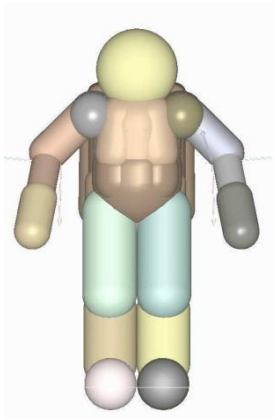


Fig. 4. Arrangement of the columns and the spheres



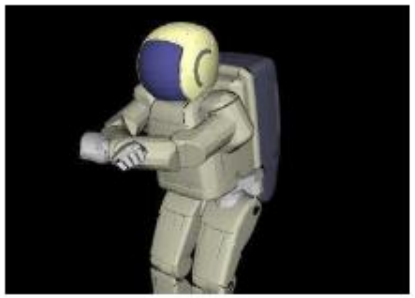


Fig. 5. Example of arm avoidance. The motion without the collision avoidance on the simulator and with on the robot.

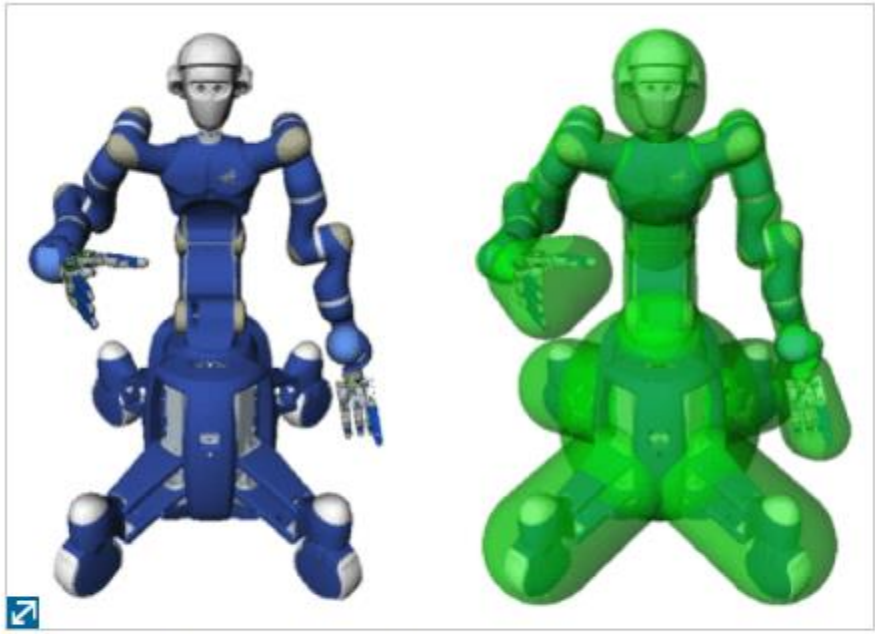


Fig. 2: Self-collision avoidance model of DLR's humanoid Justin consisting of 28 convex hulls (left arm: 8, right arm: 8, mobile platform: 5, torso: 4, head: 2, floor: 1)

View All

Presentation

1. Introduction
2. System Configuration
3. Collision avoidance control
4. Experiment
5. Conclusion

System Configuration

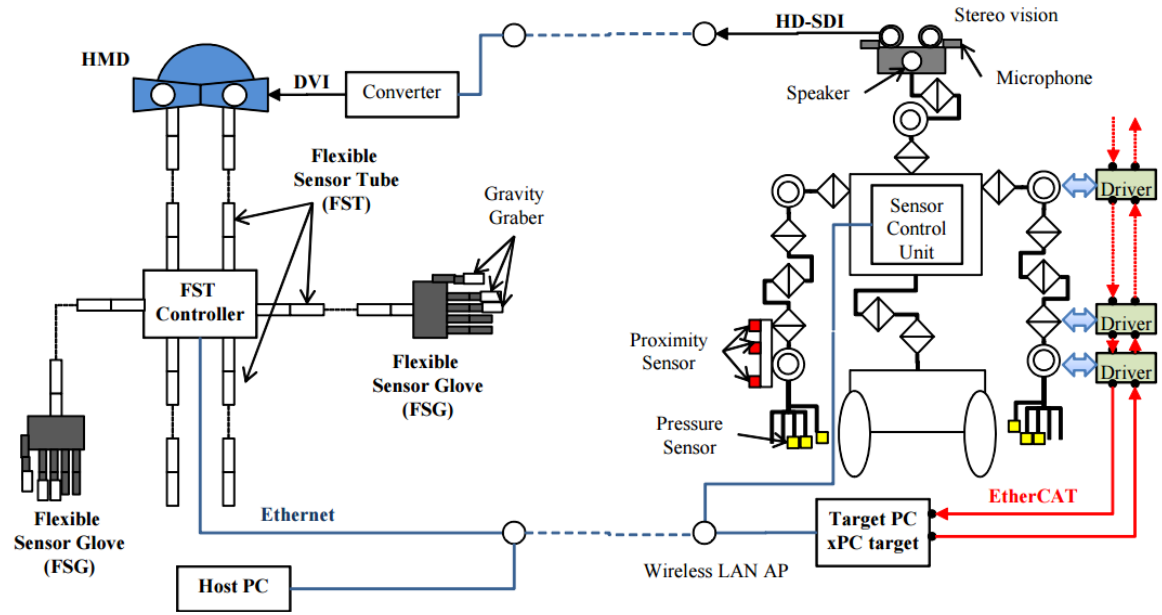


Fig. 1 System configuration

i-MarioNET : Remote Operated Humanoid Robot System Controlled by Telexistence FST

Flexible Sensor Tube

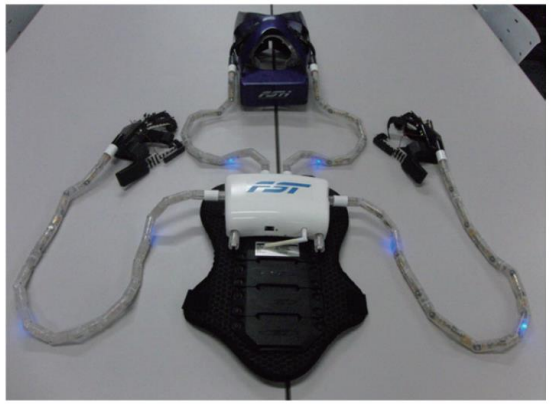
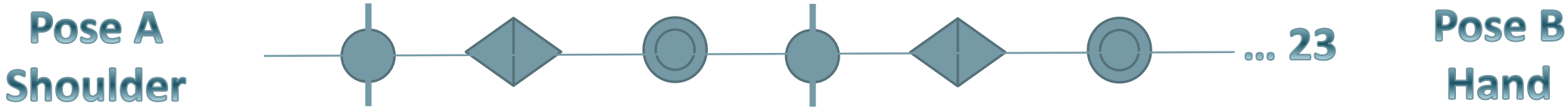


Fig. 2 Telexistence FST

Fig 3. Serie of rotational and bending joints



Slave Robot Arm 7DOF

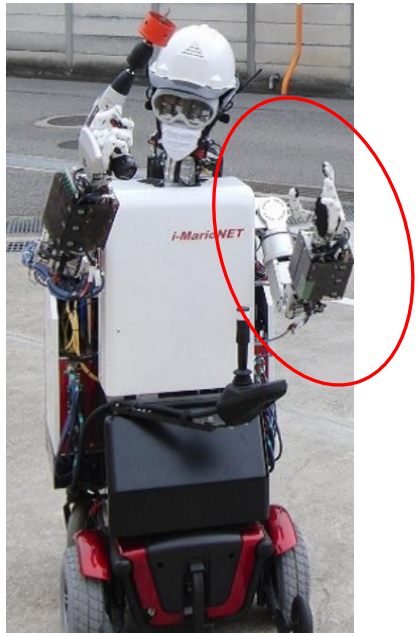


Fig 1. Slave robot

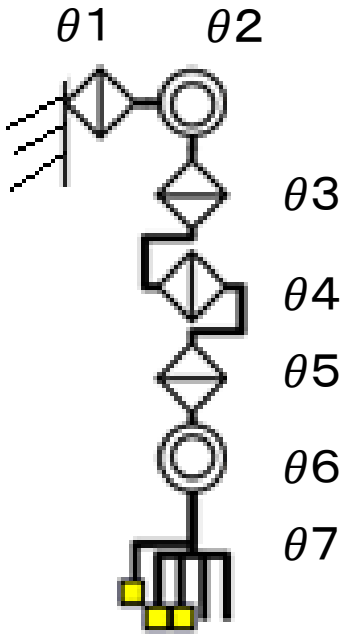


Fig 2. Model of left arm

Redundant manipulator

The kinematic redundancy is characterized by additional degrees of freedom (DOFs) with respect to those required to execute a given task.

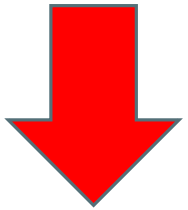
This provides means for solving sophisticated motion tasks such as

- Avoiding obstacles
- Avoiding singularities
- Optimizing manipulability
- Minimizing joint torques

L. Sciavicco and B. Siciliano, "Modelling and Control of Robot Manipulators (Advanced Textbooks in Control and Signal Processing)", *Advanced Textbooks in Control and Signal Processing*, 2005.

When does Self Collision occur?

- When any segment of the humanoid collides with another segment while the robot is in motion
- Statically when the desired pose of the robot would result in the intersection of two segments
- Structural differences



Collision not only impede the robot's motion but can cause damage to the robot itself

COLLISION AVOIDANCE CONTROL USING NULL SPACE CRITERIA Mariana FRATU, Aurel FRATU Transilvania University of Brasov, Romania

Arm modeling / Direct Kinematics

Homogeneous Transform

$$T_1(\theta_1) = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$T_2(\theta_2) = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ -\sin\theta_2 & -\cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$T_3(\theta_3) = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$T_4(\theta_4) = \begin{pmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & a_3 \\ 0 & 0 & -1 & -d_4 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

....

$$ {}^0_7T = {}^0_1T \cdot {}^1_2T \ldots {}^6_7T $$

$$ = \begin{pmatrix} {}^0_ix & {}^0_iy & {}^0_iz & {}^0_ip \\ 0 & 0 & 0 & 0 \end{pmatrix} $$

Real-time Inverse Kinematics Computation

Solved by using Levenberg-Marquardt (LM) Method , and the Collision avoidance is solved by modeling barriers in the robot Weighted Least Norm (WLN) Method.

2015 IEEE/SICE International Symposium on System Integration (SII) December 11-13, 2015. Meijo University, Nagoya, Japan

Arm Inverse Kinematics LM Method

1. Calculate de the difference between the desired position and actual

$$\mathrm{e(q)} = \begin{bmatrix} p_d - p(q) \\ \alpha(R_d R(q)^T) \end{bmatrix}$$

p_d is the desired position
p(q) is the robot actual position

2. Try to solve the equation (Newton-Raphson Method)

$$q_{k_+1} = q_k + J_k^{-1} e_k$$

where e(q)=0

The above formulation implies the following three
assumptions:
1) The number of constraints and the DOF of the robot are the same,
i.e., $n = 3m$.
2) $\boldsymbol{J}k$ is always regular.
3) Equation (3) is solvable.

Solvability-Unconcerned Inverse Kinematics by the Levenberg–Marquardt Method
Tomomichi Sugihara*, Member, IEEE*

Arm Inverse Kinematics LM Method

3. Since is a redundant manipulator the Jacobian matrix is [6x7] (Not square)

$$\mathrm{E}(\mathrm{q}) = \frac{1}{2} e^T W_E e \rightarrow min$$

So it becomes a minimization problem

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \boldsymbol{H}_k^{-1} \boldsymbol{g}_k$$

$$\boldsymbol{H}_k \equiv \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{W}_E \boldsymbol{J}_k + \boldsymbol{W}_N$$

Where

$$\boldsymbol{g}_k \equiv \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{W}_E \boldsymbol{e}_k .$$

And

W_E is the weighting matrix in order to reflect the priority level of each constraint
W_N is called the damping factor.

Solvability-Unconcerned Inverse Kinematics by the Levenberg–Marquardt Method
Tomomichi Sugihara*, Member, IEEE*

Arm inverse Kinematics Weighted Least Norm (WLN)

Compare every new calculated angle with the upper and lower limits, prevents the robot from reaching this positions.

if $q_{k+1,j} > q_{max}$ & $\Delta q_{,j}$ >0

$\Delta q_{,j}$=0

$else\ if$ $q_{k+1,j} < q_{min}$ & $\Delta q_{,j}$ <0

$\Delta q_{,j}$=0

q_j	$q_{max,j}[degree]$	$q_{\min,j}[degree]$
q_1	0	-90
q_2	0	-90
q_3	180	0
q_4	100	0
q_5	90	-90
q_6	-80	-100
q_7	30	-30

Presentation

1. Introduction
2. System Configuration
3. Collision avoidance control
4. Experiment
5. Conclusion

Previous Method

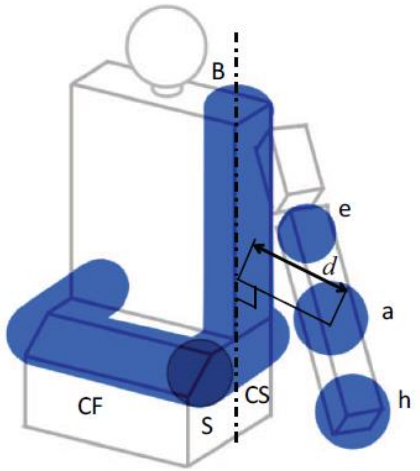


Fig. 4. Arrangement of the columns and the spheres

Arrange of mobile sphere against static cylinders
Where d is the distance between two objects and

$$\Delta d = \boldsymbol{J}_{\text{avoid}} \Delta \boldsymbol{q} \tag{1}$$

where $\boldsymbol{J}_{\text{avoid}}$ is the Jacobian matrix. By solving it,

$$\Delta \boldsymbol{q} = \boldsymbol{J}^{\dagger}_{\text{avoid}} \Delta d + (\boldsymbol{I} - \boldsymbol{J}^{\dagger}_{\text{avoid}} \boldsymbol{J}_{\text{avoid}}) \boldsymbol{k} \tag{2}$$

where $\boldsymbol{k}$ in R^7 is an arbitrary vector, and $\boldsymbol{J}^{\dagger}_{\text{avoid}}$ is the Moore-Penrose pseudo inverse matrix. This equation is utilized to generate the avoidance motions as follows.

$$\begin{aligned} \Delta \boldsymbol{q}'_k &= -\lambda(d_{k+1}) \boldsymbol{J}^{\dagger}_{\text{avoid},k=0} \Delta d_k \\ &+ (\boldsymbol{I} - \lambda(d_{k+1}) \boldsymbol{J}^{\dagger}_{\text{avoid},k=0} \boldsymbol{J}_{\text{avoid},k=0}) \Delta \boldsymbol{q}_k \end{aligned} \tag{3}$$

Operation Assist for a Teleoperated Robot System Controlled with a Lightweight and High-Operable Master Device Yosuke
Matsumoto, Akio Namiki, and Kenta Negishi

Previous Method

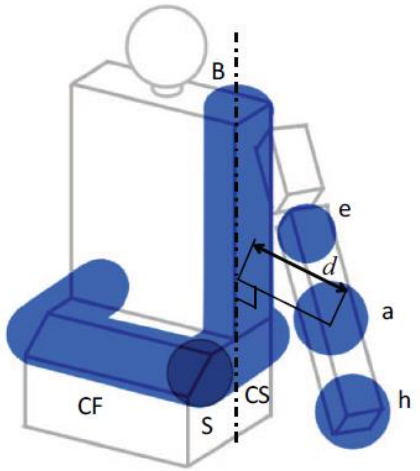


Fig. 4. Arrangement of the columns and the spheres

where k is the number of iterations, and Δq_k is the joint angles generated without considering the self-collision avoidance, and $\lambda(d)$ is the parameter to switch tasks defined as

$$\lambda(d) = \begin{cases} 0 & \text{if } \Delta d > 0 \\ \left(\frac{d_m}{d}\right)^n, n = 1, 2, 3, \cdots & \text{elseif } d \geq d_m \\ 1 & \text{else} \end{cases} \quad (4)$$

where d_m is the boundary for self-collision avoidance, and n is a suitable positive number, which is set to 5 in this study.

Operation Assist for a Teleoperated Robot System Controlled with a Lightweight and High-Operable Master Device Yosuke Matsumoto, Akio Namiki, and Kenta Negishi

Previous Method Disadvantages

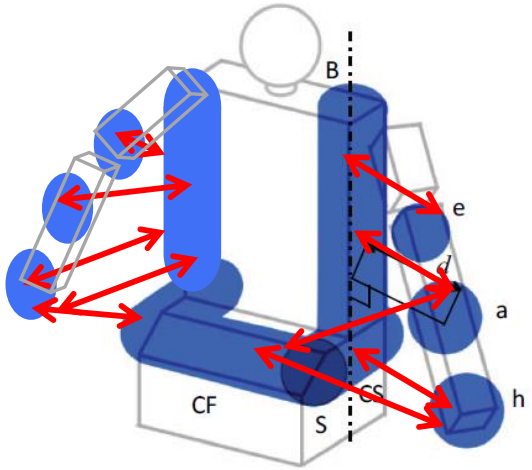
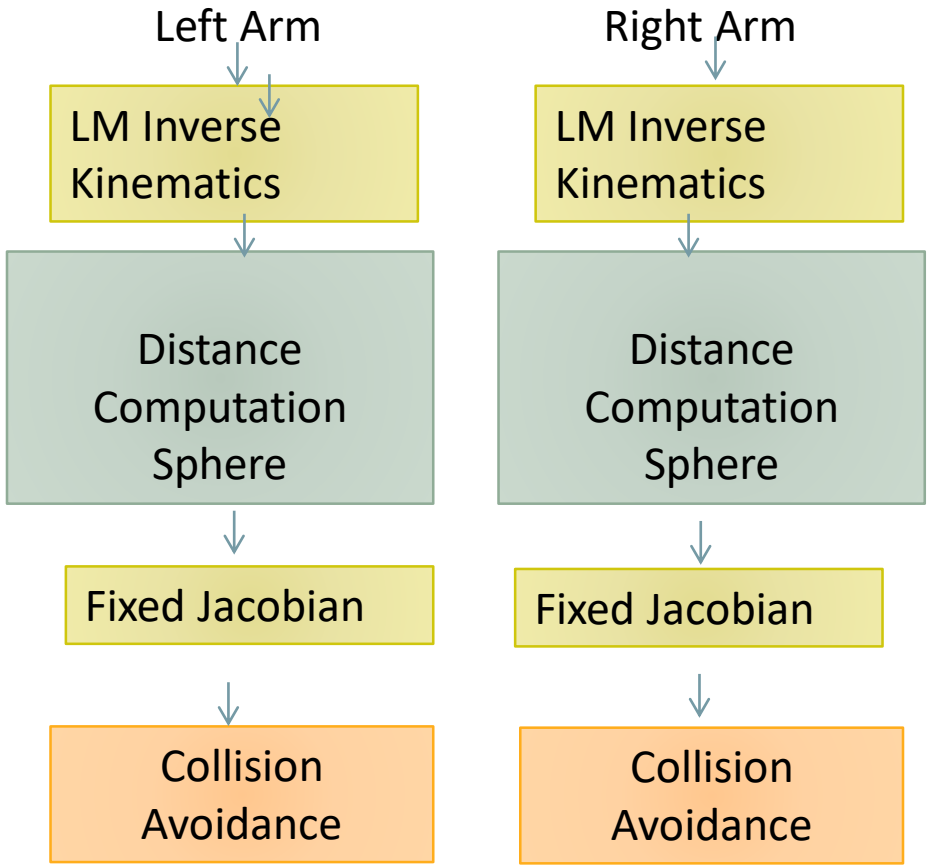


Fig. 4. Arrangement of the columns and the spheres

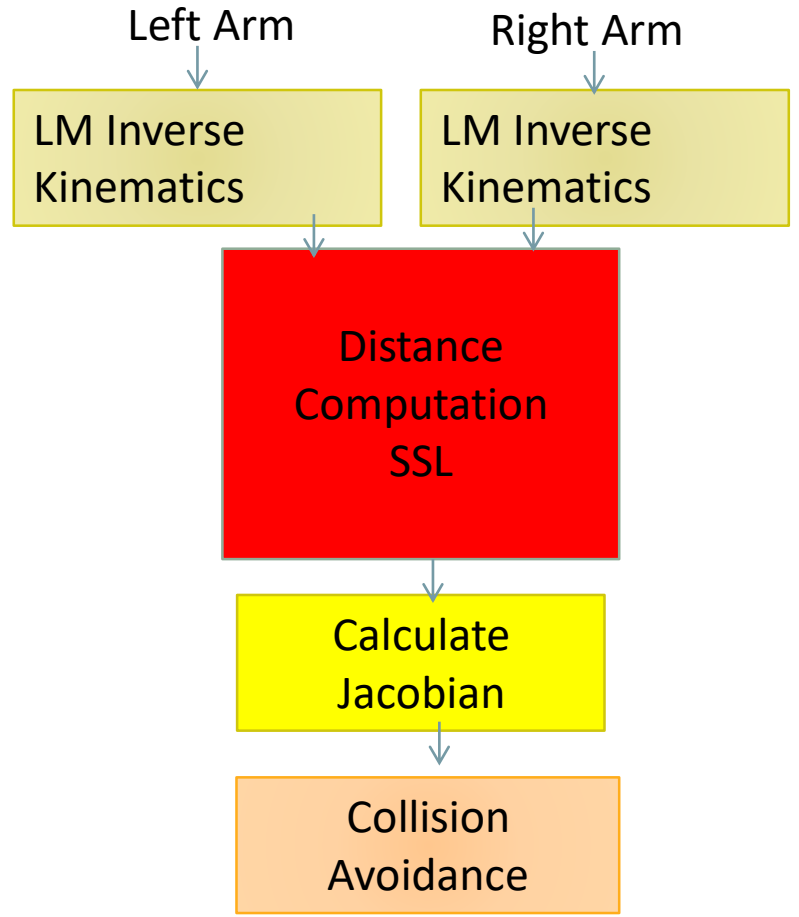
- The distance each object has to be calculated
- The Jacobian matrix was previously fixed, which doesn't allow changes in the coordinate system
- Calculates every arm separately so both arms can still collide
- High Computational cost
- Low flexibility to changes

Operation Assist for a Teleoperated Robot System Controlled with a Lightweight and High-Operable Master Device Yosuke Matsumoto, Akio Namiki, and Kenta Negishi

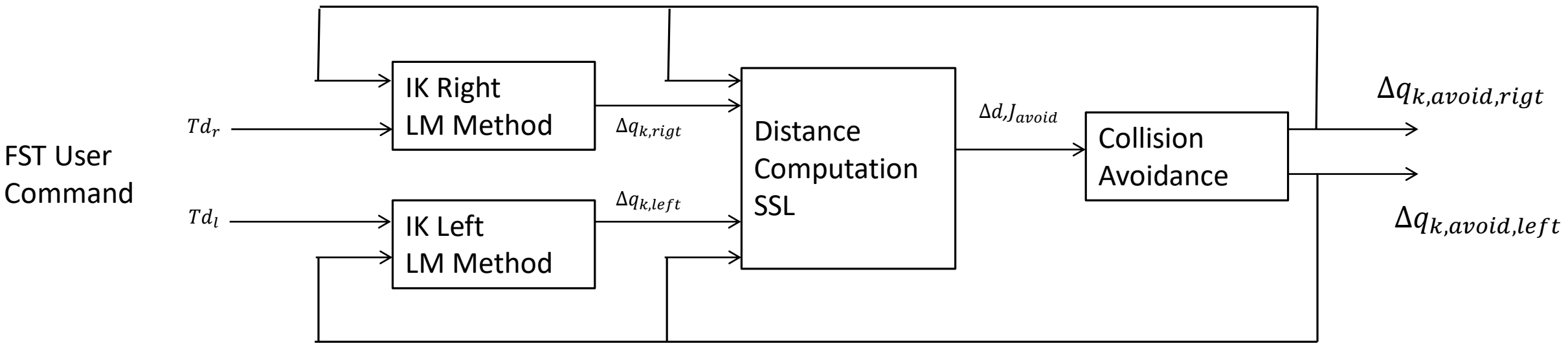
Previous Method



Proposed Method

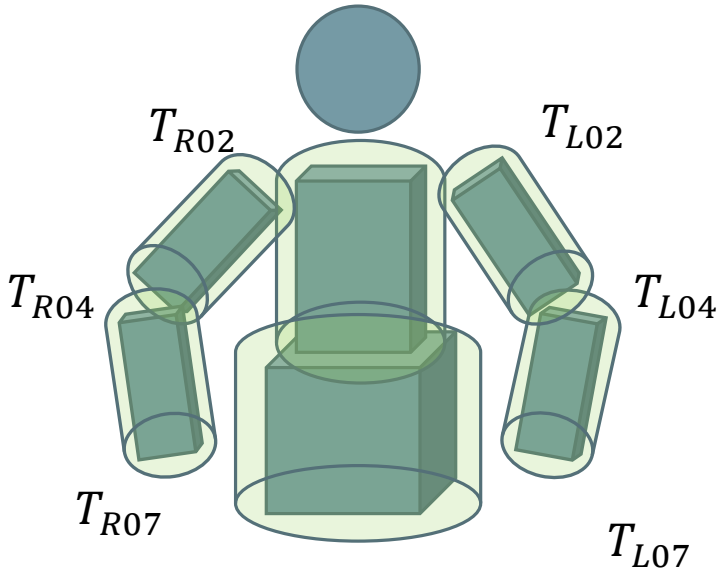


Control Diagram



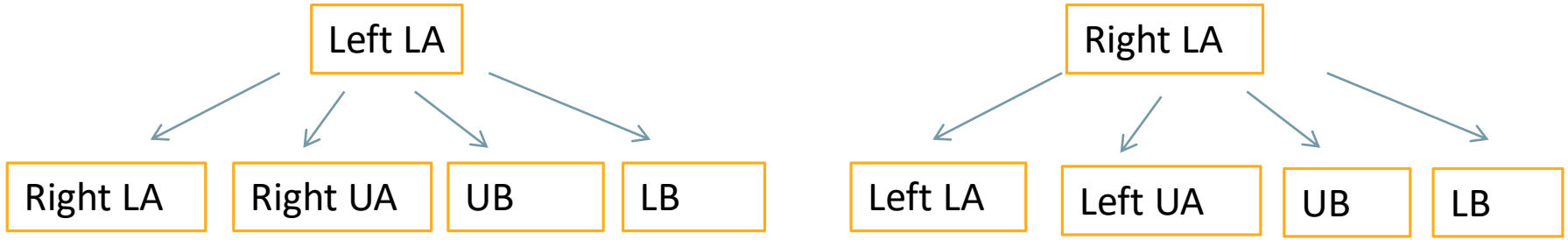
Distance Calculation Modeling

Direct Kinematics



- 6 Capsules
  - 2 Lower Arm
  - 2 Upper Arm
  - Upper Body
  - Lower Body

Distance Calculation Collision Tree



LOWER ARM - LA
UPPER ARM - UP
LOWER BODY - LB
UPPER BODY - UB

Sweep Sphere Line

$$C_1(s) = P_1 + s\boldsymbol{d_1}, \qquad \boldsymbol{d_1} = Q_1 - P_1, \qquad 0 \le s \le 1$$

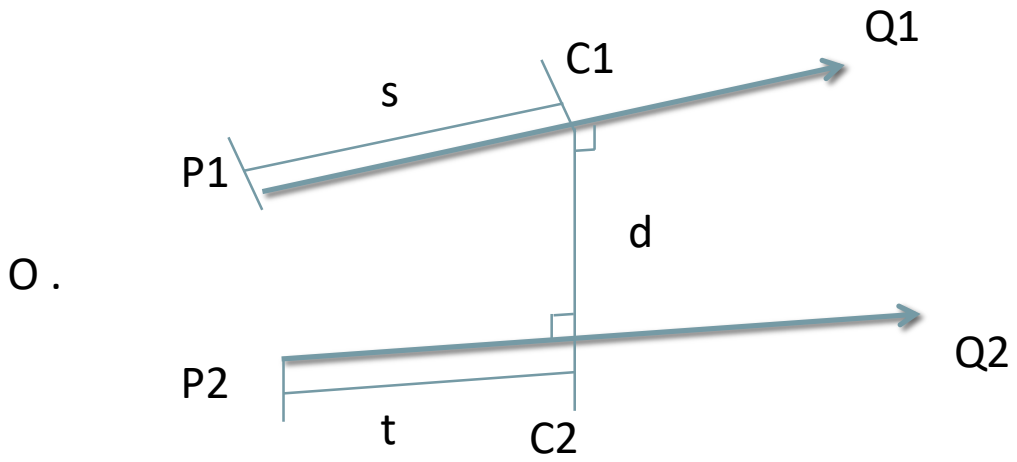
$$C_2(t) = P_2 + t\boldsymbol{d_2}, \qquad \boldsymbol{d_2} = Q_2 - P_2, \qquad 0 \le t \le 1$$

$$d = \|C1 - C2\| - R1 - R2$$

$$u_0 = \frac{C1 - C2}{\|C1 - C2\|}$$

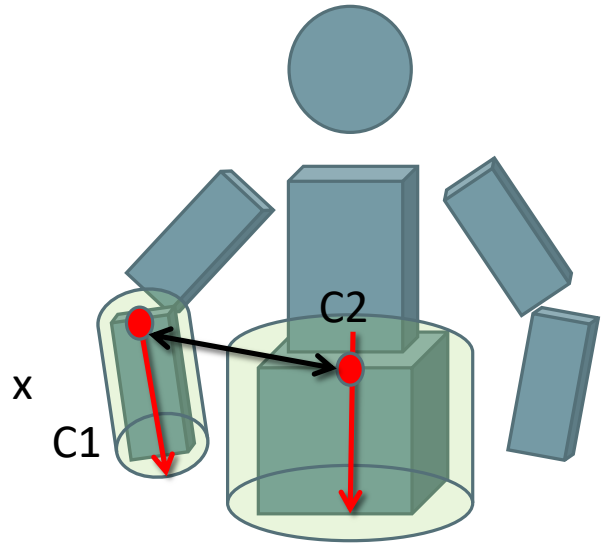
$$s = (P_2 + t\boldsymbol{d_2} - P_1) * \frac{d_1}{d_1} * d_1$$

$$t = (P_1 + s\boldsymbol{d_1} - P_2) * \frac{d_2}{d_2} * d_2$$



R = Capsule Radius

Jacobian for avoidance
Example Left LA vs Lower Body



$$Jc(q) = \begin{bmatrix} {}^{0}z1\ x\ \ p_{x,1} & {}^{0}z2\ x\ \ p_{x,2} & \cdots & {}^{0}z1\ x\ \ p_{x,1} \\ {}^{0}z1 & {}^{0}z2 & \cdots & {}^{0}z7 \end{bmatrix}$$

$$J_{avoid}(q) = -u^{T}\ .Jcx(q)$$

$-u$ = Unit Vector in the direction of the Collision

Jcx = Jacobian Matrix in the Cartesian Space

It is only necessary to calculate the Jacobian matrix of the point closest to collision, instead of 3 spheres before

Limit Control

$$\begin{aligned}\Delta \boldsymbol{q}'_k &= -\lambda(d_{k+1})\boldsymbol{J}^{\dagger}_{\text{avoid},k=0}\Delta d_k \\ &+ (\boldsymbol{I} - \lambda(d_{k+1})\boldsymbol{J}^{\dagger}_{\text{avoid},k=0}\boldsymbol{J}_{\text{avoid},k=0})\Delta \boldsymbol{q}_k\end{aligned}$$

As the FST comes closer to the collision the arms moves farther away, even when the collision has already been prevented, therefore

if$|\Delta d_k| > \left|\Delta d_{k,rate}\right|$

$|\Delta d_k| = \left|\Delta d_{k,rate}\right|$

end

Presentation

1. Introduction
2. System Configuration
3. Collision avoidance control
4. Experiment
5. Conclusion

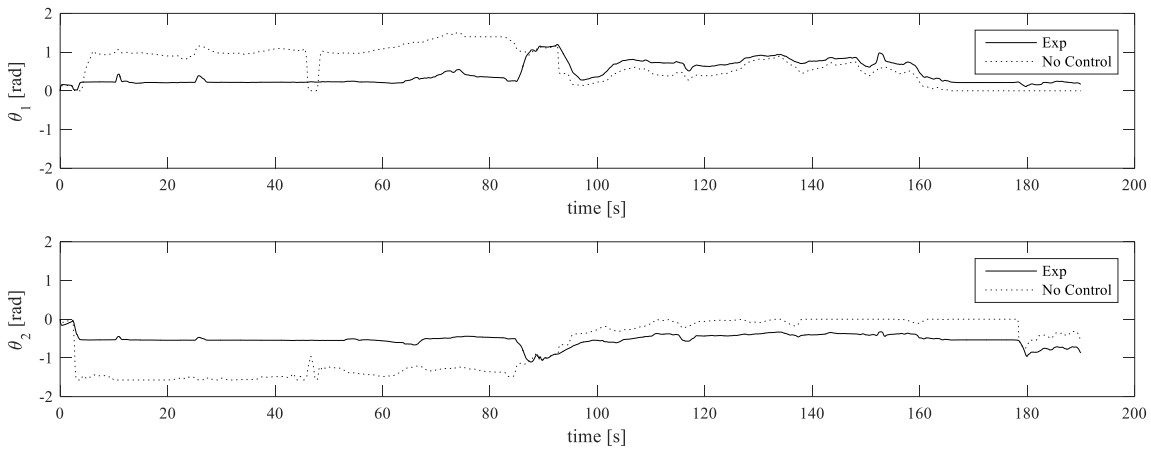
Experiment

1. Move left arm towards body
2. Move right arm towards body
3. Move both arms towards each other

Experiment

Video

Results Angles



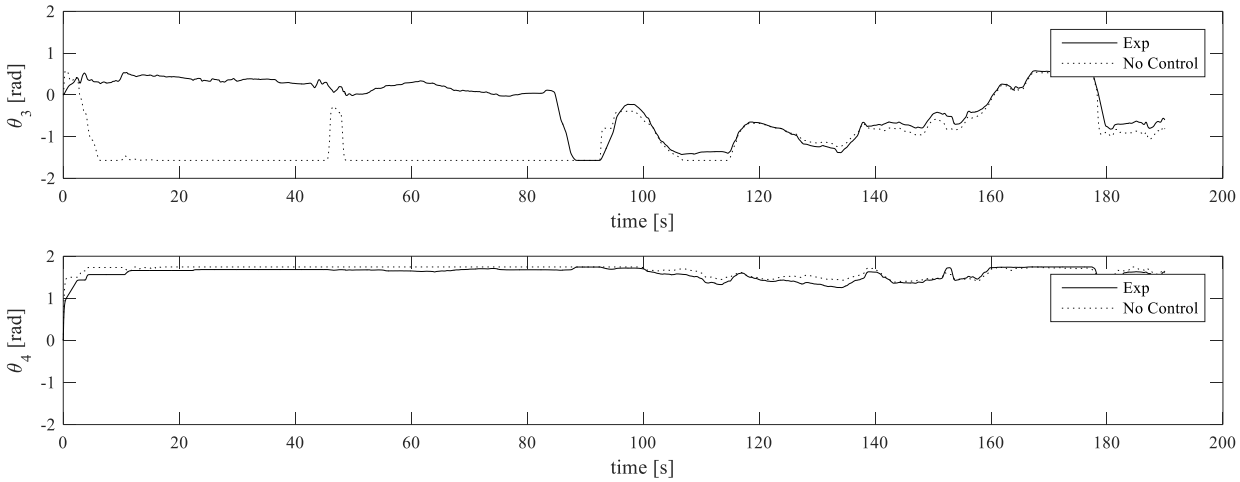


Fig 1. Right Arm Angles

Results Angles

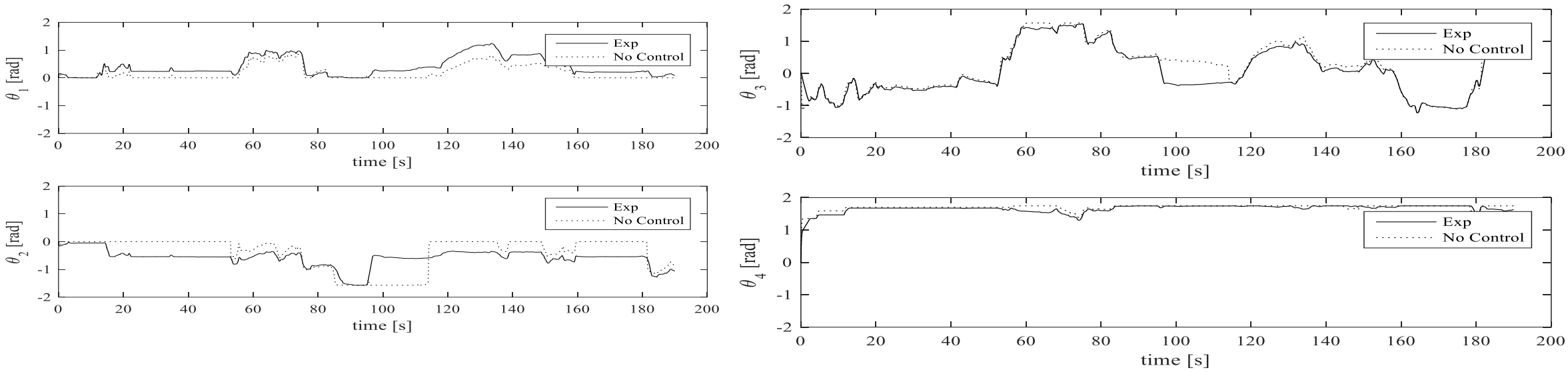


Fig 2. Left Arm Angles

Result Distance to Objects

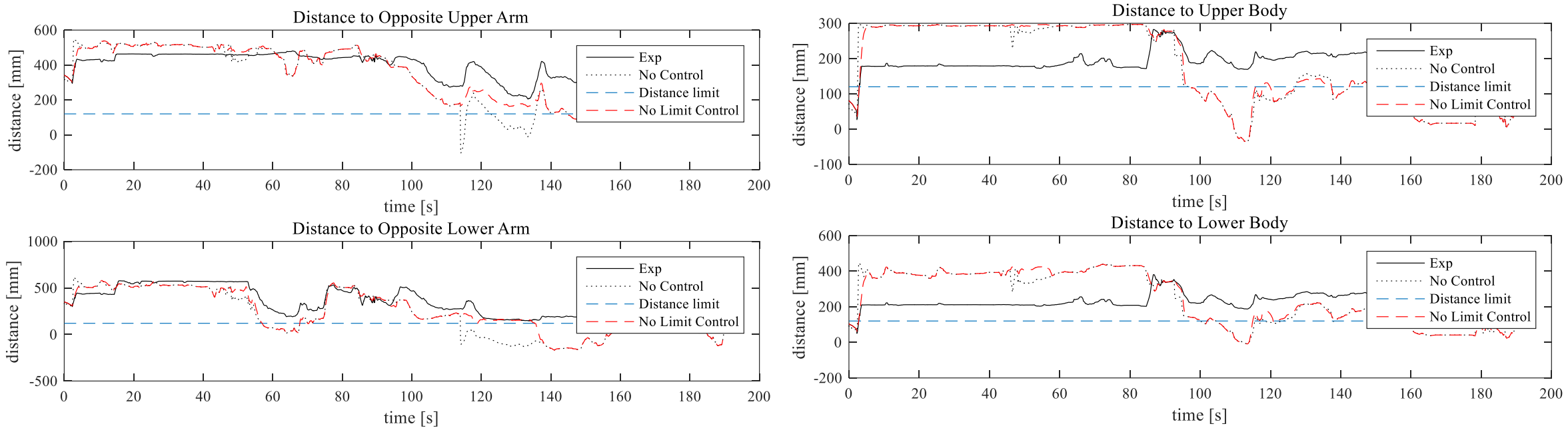
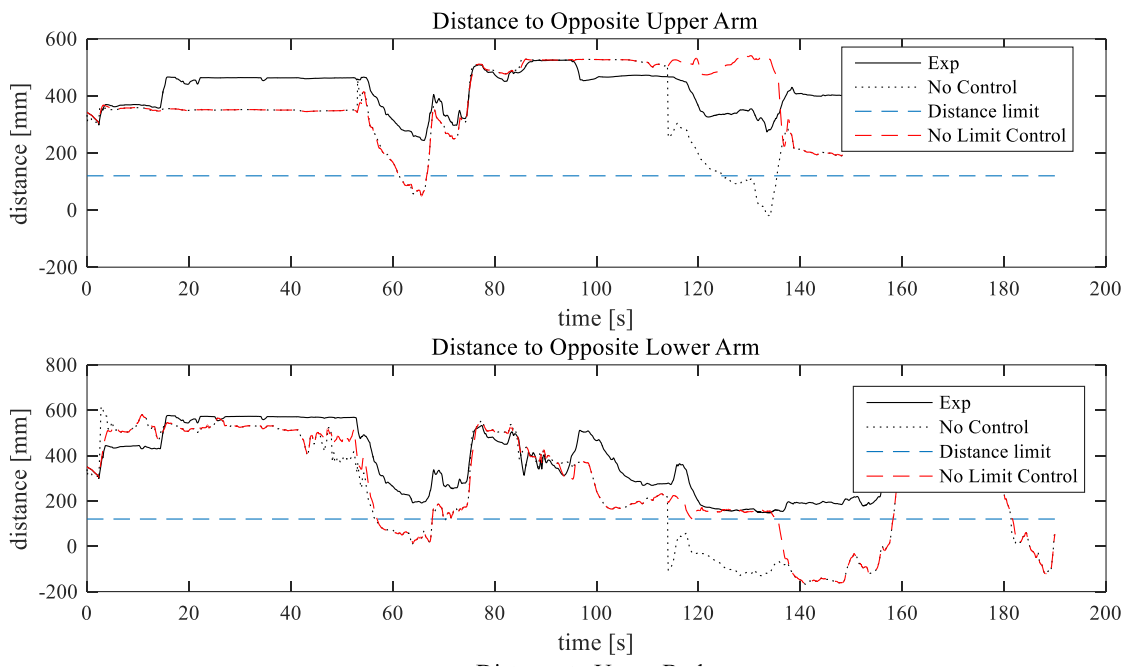


Fig 1. Lower Right Arm to Objects Distance



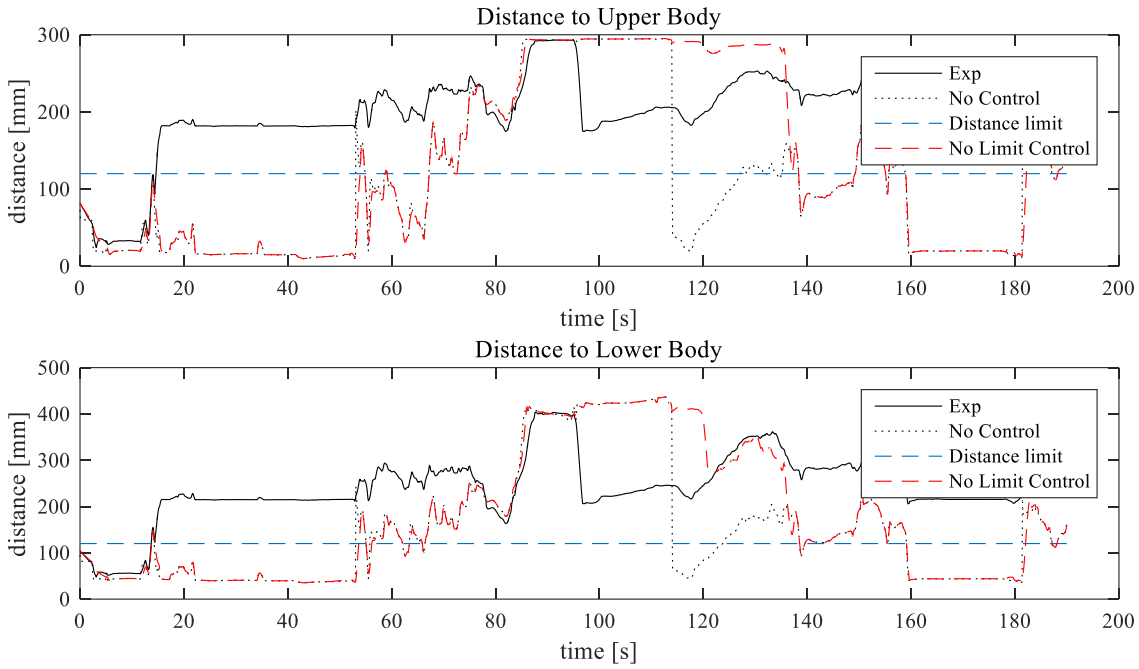


Fig 2. Lower Left Arm to Objects Distance

Conclusions

1. It is necessary to match the position and the orientation of the FST with the “end effector” so the algorithm works properly.

2. The initial position is also considered as collision position so it is also necessary to change that

3. The parameters can change so the arms can move even closer to each other

4. It is necessary to update the program to the actual coordinate system, but since the Jacobian matrix is calculated each time, it will be easier.

5. However the algorithm was effective in avoiding the upper body and both arms, in contrast with the previous method after the initial collision, no more impacts where computed.

Thanks for your attention