MEAM 520

The PUMA 260

Katherine J. Kuchenbecker, Ph.D.

General Robotics, Automation, Sensing, and Perception Lab (GRASP)

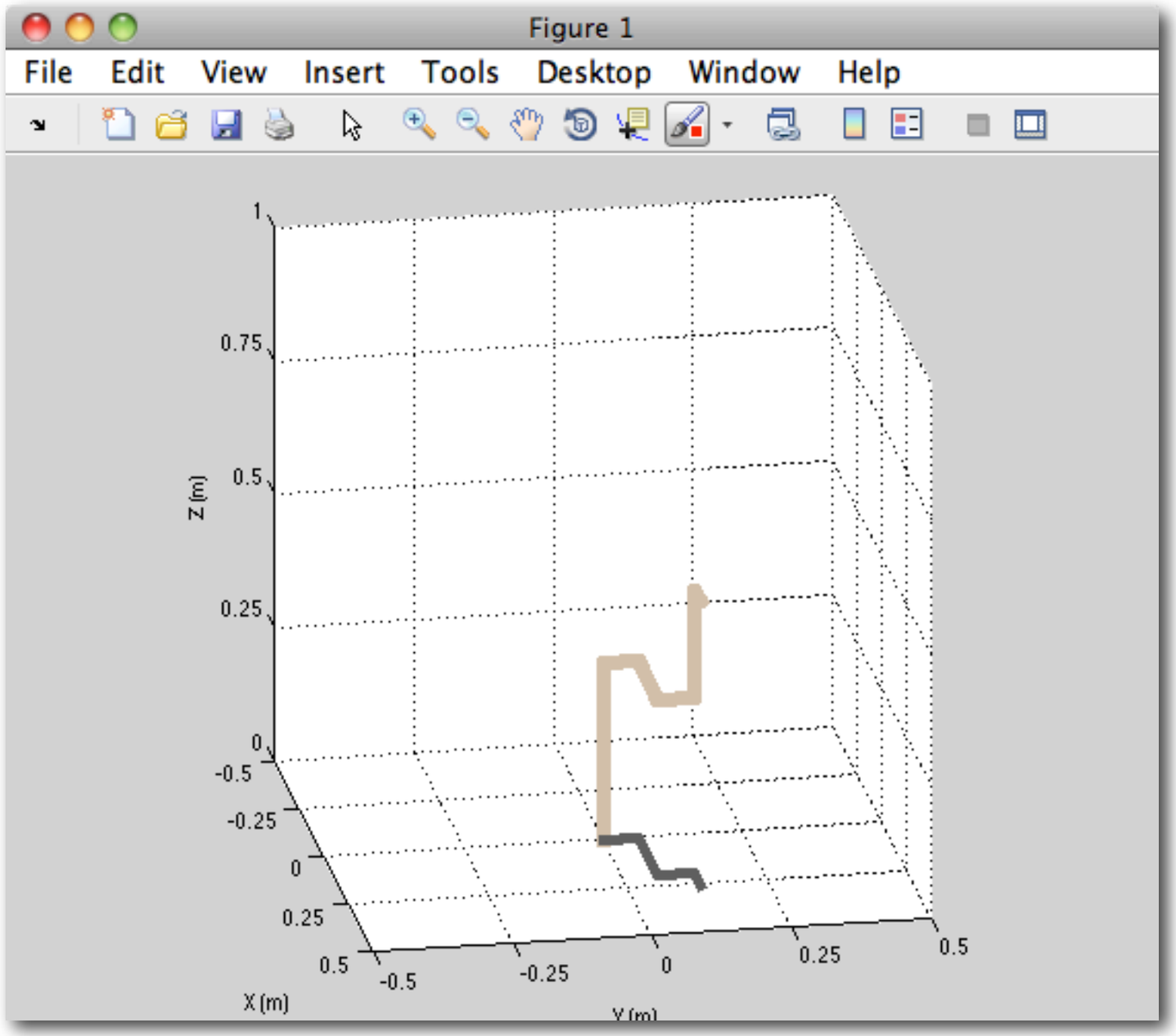
MEAM Department, SEAS, University of Pennsylvania

GRASP LABORATORY

Lecture 16: October 24, 2013

Project 1

PUMA Dance



- I have now checked all of the Project 1 submissions.

- A few teams were missing files. I have requested those to be resubmitted, and I saw several come in. I'll recheck tonight.

- A few teams used pumaMove() instead of pumaServo(). Only pumaServo() works on the real robot.

- I fixed two lines in every team1xx_puma_dance.m file to enable running on the real robot.

MEAM 520

https://piazza.com/class/hf935b0sz1m5r3?cid=164

piazza | MEAM 520 | Q & A | Course Page | Manage Class | Katherine J. Kuchenbecker

hw1 hw2 hw3 hw4 hw5 hw6 hw8 final_exam lecture11 lecture12 lecture13 lecture14 project1 midterm_exam other office_hours textbook matlab

note

stop following 36 views

Actions

Filming Your PUMA Dance

We have created an online calendar that you can use to sign up for a time to film your team's PUMA Dance:
http://penn_meam520.youcanbook.me/

At present, there is only one slot listed - today (Wednesday) at 3:00 p.m. by Samarth – but the TAs will soon be adding other slots. Please confer with your partner to find a time that works for both of you and then sign up. It would be great if we can film all of the dances before the midterm on November 5.

The filming will take place in B2 Towne. This room is on the basement level of the Towne Building, at the bottom of the southwest stairwell, just inside the door there.

Please arrive on time, since there is a lot to accomplish. You don't need to bring anything. Your team's code will already be loaded on the PUMA computer.

Happy PUMA dancing!

project1 puma

edit · good note | 0

1 day ago by Katherine J. Kuchenbecker

followup discussions *for lingering questions and comments*

Start a new followup discussion

Compose a new followup discussion

Average Response Time: 9 min

Special Mentions: Eduardo Ed Garcia answered HW6 Q1d in 18 min. 1 day ago

Online Now | This Week: 8 | 105

MEAM 520 Staff

http://penn_meam520.youcanbook.me/

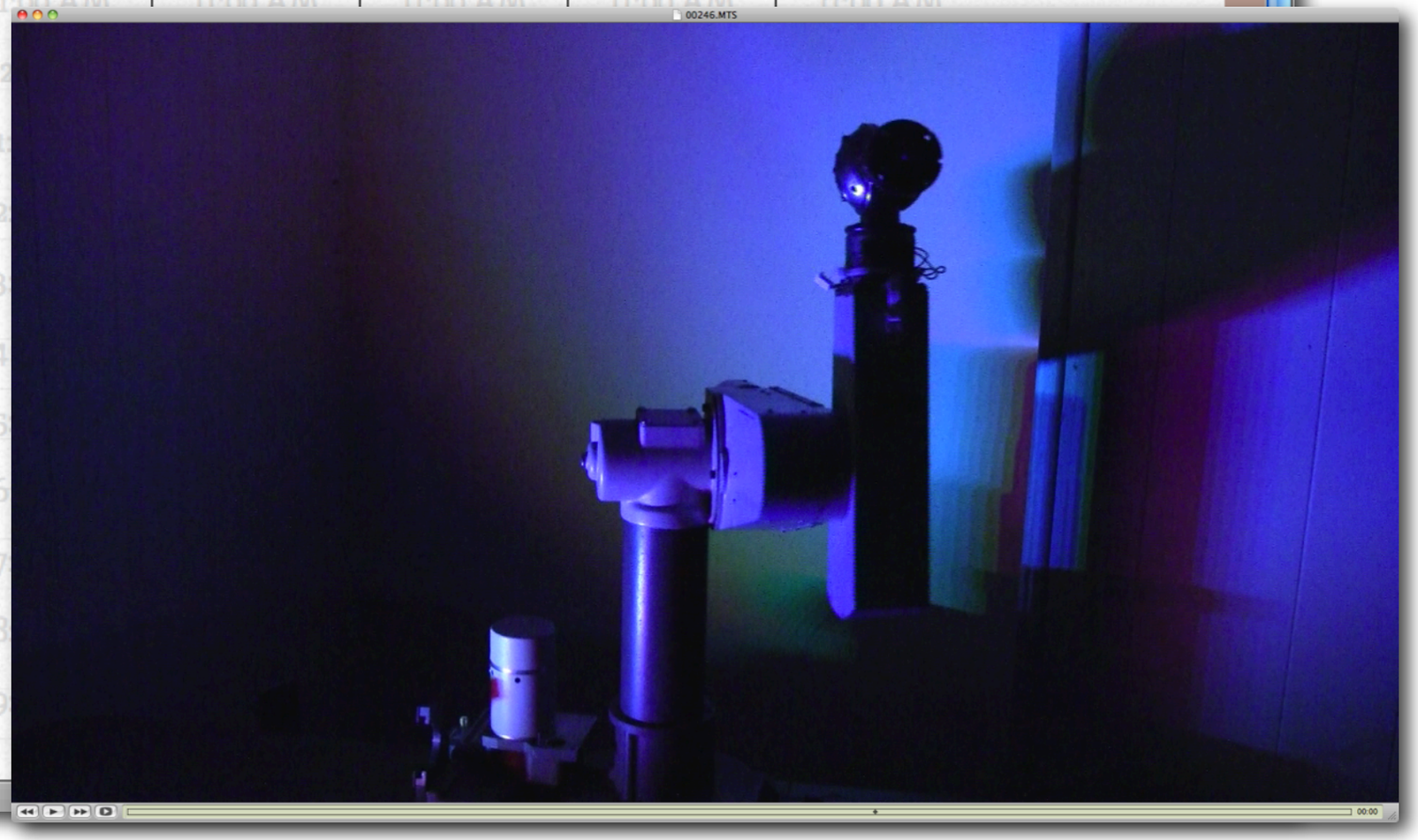
Google

MEAM 520 Staff

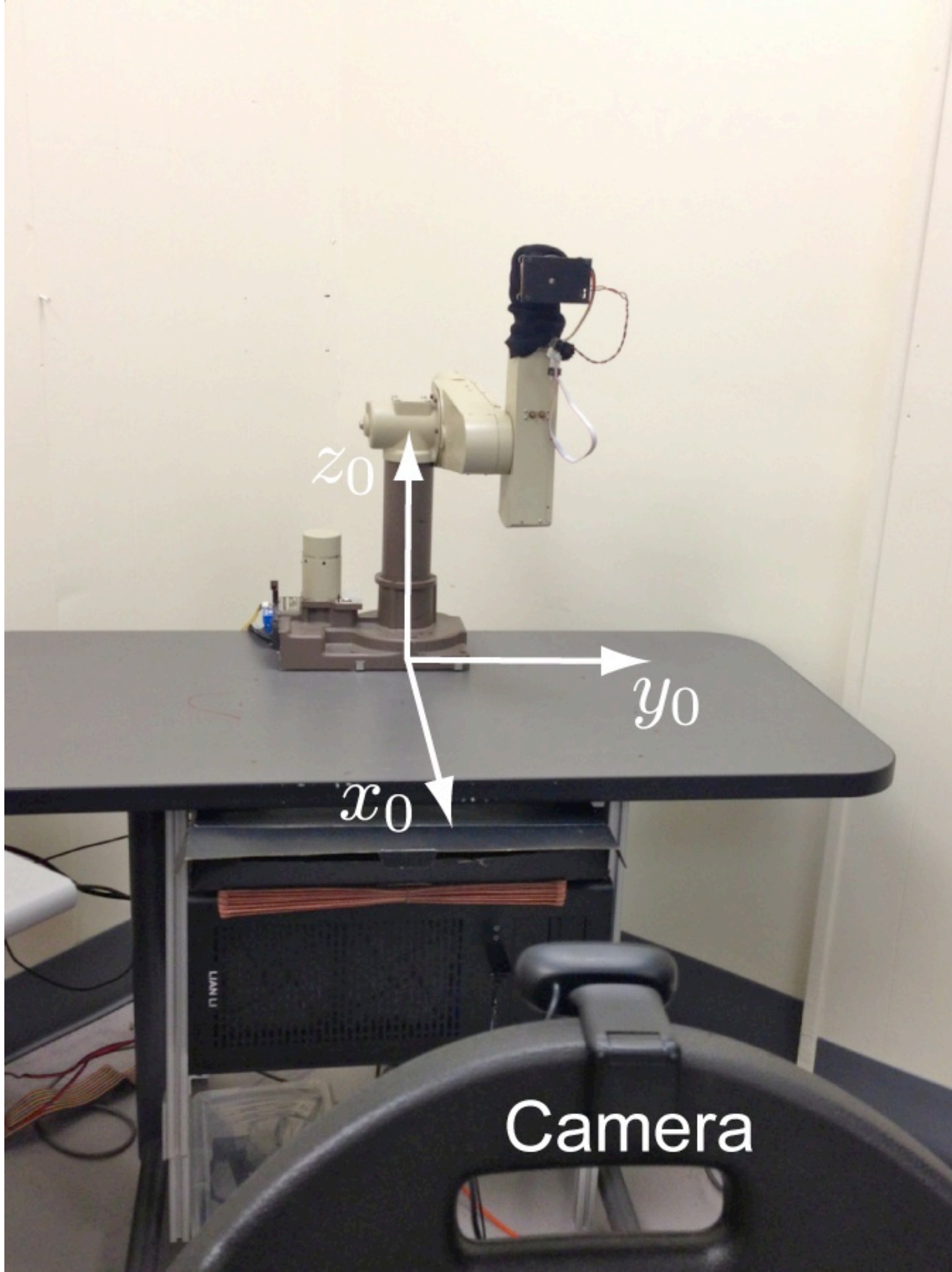
Click on any time to make a booking.

	Thu 10/24/13	Fri 10/25/13	Sat 10/26/13	Sun 10/27/13	Mon 10/28/13	Tue 10/29/13	Wed 10/30/13
10:00 AM							
11:00 AM							
2:00 PM							
3:00 PM							
4:00 PM							
5:00 PM							
6:00 PM							
7:00 PM							
8:00 PM							
9:00 PM							

Please sign up for a slot.




EMERGENCY
STOP
BACO


z_0
y_0
x_0
Camera

MEAM 520 Calendar

Ongoing – Film PUMA Dance

Tuesday 10/22 – Almost Finish Chapter 4
Thursday 10/24 – Finish Chapter 4,
Introduce HW 7 and
Discuss PUMA
Sunday 10/27 – Homework 6 Due

Tuesday 10/29 – Start Chapter 6
Thursday 10/31 – Finish Chapter 6
Sunday 11/3 – Homework 7 Due

Tuesday 11/5 – Midterm Exam
Thursday 11/7 – Start Chapter 3.3 (IK)

Also planning a review session....

Homework 6:
Velocity Kinematics and Jacobians

MEAM 520, University of Pennsylvania
Katherine J. Kuchenbecker, Ph.D.

October 17, 2013

This paper-based assignment is due on **Sunday, October 27, by midnight (extended)** to the bin outside Professor Kuchenbecker's office, Towne 224. Late submissions will be accepted until Wednesday, October 30, by midnight (11:59:59 p.m.), but they will be penalized by 10% for each partial or full day late, up to 30%. After the late deadline, no further assignments may be submitted.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you write down must be your own work, not copied from any other individual or a solution manual. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. If you get stuck, post a question on Piazza or go to office hours!

These problems are loosely based on problems that appear in the printed version of the textbook, *Robot Modeling and Control* by Spong, Hutchinson, and Vidyasagar (SHV); all of the needed instructions are included in this document. Write in pencil, show your work clearly, box your answers, and staple together all pages of your assignment. This assignment is worth a total of 20 points.

1. Skew-Symmetric Matrices *(6 points)*

a. We define $\hat{u} = [x \;\; y \;\; z]^T$ to be a unit vector. What is $S(\hat{u})$, the skew-symmetric matrix associated with this unit vector?

b. Now define $\vec{v} = [0 \;\; 10 \;\; 0]^T$ and calculate $S(\hat{u})\vec{v}$.

c. What is the geometric meaning of the result you obtained in the last step? Draw a sketch with an arbitrarily chosen unit vector $\hat{u}$ to explain. Think about both the magnitude and the direction of the result.

d. Show that $S^3(\hat{u}) = -S(\hat{u})$.

e. What is the geometric meaning of the equation $S^3(\hat{u}) = -S(\hat{u})$? Explain using words and a sketch.

f. $R_{\hat{u},\theta}$ is a rotation matrix representing rotation by the time-varying angle θ about the constant unit vector $\hat{u}$. By considering equation (2.43) in the book, one can show that $R_{\hat{u},\theta} = I + S(\hat{u})\sin\theta + S^2(\hat{u})\text{vers}\,\theta$, where the versine $\text{vers}\,\theta = 1 - \cos\theta$. Note that you do not need to show this equivalence. Instead, use this equivalence and the equation from the previous step to show that $\frac{dR_{\hat{u},\theta}}{d\theta} = S(\hat{u})R_{\hat{u},\theta}$.

g. What is the intuitive meaning of the equation $\frac{dR_{\hat{u},\theta}}{d\theta} = S(\hat{u})R_{\hat{u},\theta}$?

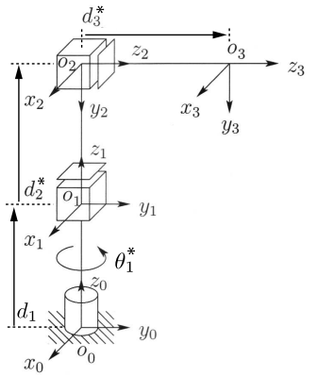
Homework 6

Written Assignment on Velocity Kinematics, Jacobians, and Singularities

Due Sunday 10/27 (extended)

Any questions?

2. Three-link Cylindrical Manipulator ***(7 points)***



$$T_3^0 = \begin{bmatrix} c_1 & 0 & -s_1 & -s_1 d_3 \\ s_1 & 0 & c_1 & c_1 d_3 \\ 0 & -1 & 0 & d_1 + d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Above are the DH diagram and the corresponding transformation matrix T_3^0 for a three-link cylindrical manipulator. The diagram shows θ_1 at zero, and d_2 and d_3 are shown at positive displacements. These materials are adapted from the derivation on pages 85 and 86 of the book.

a. Use the position of the end-effector in the base frame to calculate the 3×3 linear velocity Jacobian J_v for this robot.

b. Use the positions of the origins o_i and the orientations of the z-axes z_i to calculate the 3×3 linear velocity Jacobian J_v for the same robot. You should get the same answer as before.

c. Find the 3×3 angular velocity Jacobian J_ω for the same robot.

d. Imagine this robot is at $\theta_1 = \pi/2$ rad, $d_2 = 0.2$ m, and $d_3 = 0.3$ m, and its joint velocities are $\dot{\theta}_1 = 0.1$ rad/s, $\dot{d}_2 = 0.25$ m/s, and $\dot{d}_3 = -0.05$ m/s. What is v_3^0, the linear velocity vector of the end-effector with respect to the base frame, expressed in the base frame? Make sure to provide units with your answer.

e. For the same situation, what is ω_3^0, the angular velocity vector of the end-effector with respect to the base frame, expressed in the base frame? Make sure to provide units with your answer.

f. Use your answers from above to derive the singular configurations of the arm, if any. Here we are concerned with the linear velocity of the end-effector, not its angular velocity. Be persistent with the calculations; they should reduce to something nice.

g. Sketch the cylindrical manipulator in each singular configuration that you found, and explain what effect the singularity has on the robot's motion in that configuration.

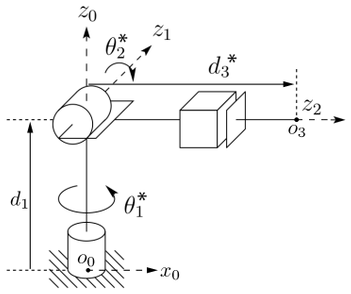
Homework 6

Written Assignment on Velocity Kinematics, Jacobians, and Singularities

Due Sunday 10/27 (extended)

Any questions?

3. Three-link Spherical Manipulator ***(7 points)***



$$d_3^0 = \begin{bmatrix} c_1 c_2 d_3 \\ s_1 c_2 d_3 \\ d_1 - s_2 d_3 \end{bmatrix}$$

Above are the DH diagram and the corresponding tip position vector d_3^0 for a three-link spherical manipulator. The diagram shows θ_1 and θ_2 at zero, and d_3 is shown at a positive displacement.

a. Calculate the 3×3 linear velocity Jacobian J_v for this manipulator. You may use any method you choose.

b. Find the 3×3 angular velocity Jacobian J_ω for the same robot.

c. Imagine this robot is at $\theta_1 = \pi/4$ rad, $\theta_2 = 0$ rad, and $d_3 = 1$ m. What is ω_3^0, the angular velocity vector of the end-effector with respect to the base frame, expressed in the base frame, as a function of the joint velocities $\dot{\theta}_1$, $\dot{\theta}_2$, and $\dot{d}_3$? Make sure to provide units for any coefficients in these equations, if needed.

d. For the same configuration described in the previous step, what is v_3^0, the linear velocity vector of the end-effector with respect to the base frame, expressed in the base frame, as a function of the joint velocities $\dot{\theta}_1$, $\dot{\theta}_2$, and $\dot{d}_3$? Provide units for any coefficients in these equations, if needed.

e. What instantaneous joint velocities should I choose if the robot is in the configuration described in the previous steps and I want its tip to move at $v_3^0 = [0 \text{ m/s} \;\; 0.5 \text{ m/s} \;\; 0.1 \text{ m/s}]^T$? Make sure to provide units with your answer.

f. Use your answers from above to derive the singular configurations of the arm, if any. Here we are concerned with the linear velocity of the end-effector, not its angular velocity. Be persistent with the calculations; they should reduce to something nice.

g. Sketch the spherical manipulator in each singular configuration that you found, and explain what effect the singularity has on the robot's motion in that configuration.

h. Would the singular configuration sketches you just drew be any different if we had chosen different positive directions for the joint coordinates? What if we had selected a different zero configuration for this robot? Explain.

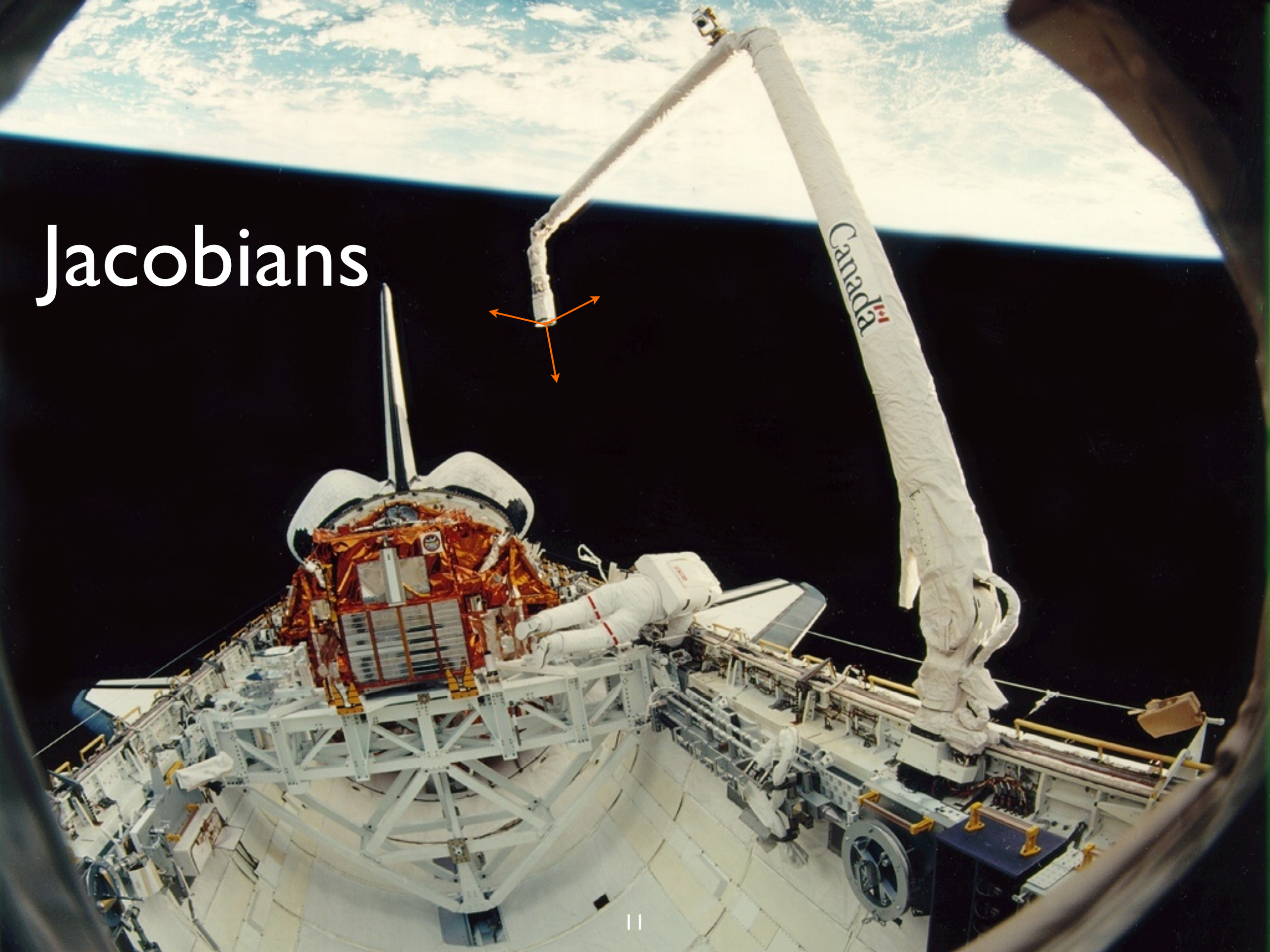
Homework 6

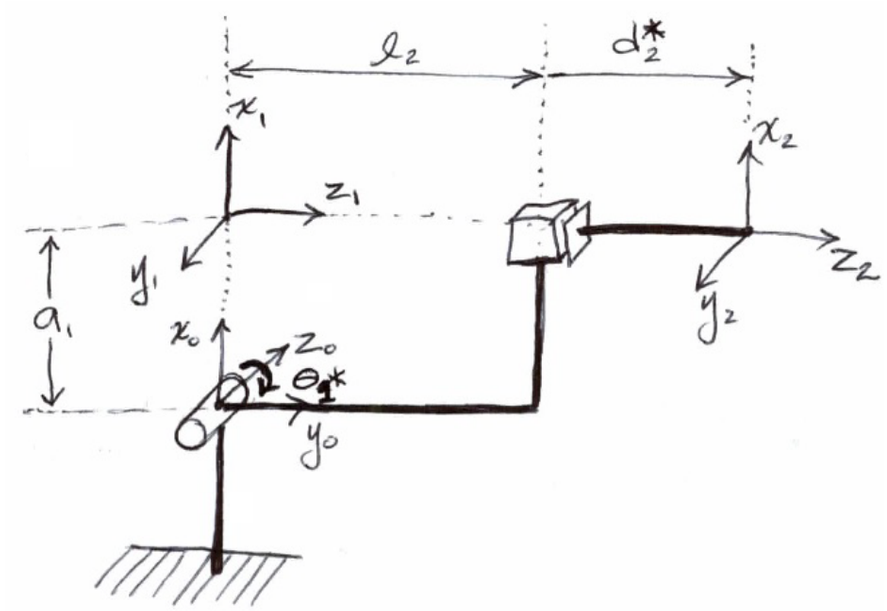
Written Assignment on Velocity Kinematics, Jacobians, and Singularities

Due Sunday 10/27 (extended)

Any questions?

Jacobians





$$\xi = J(q)\dot{q}$$

(6 x 1) body velocity

but this is not the derivative of any vector

(6 x n) Jacobian

(n x 1) joint velocities

Questions ?

$$\dot{X} = \begin{bmatrix} \dot{d} \\ \dot{\alpha} \end{bmatrix} = J_a(q)\dot{q}$$

(6 x n) Analytical Jacobian

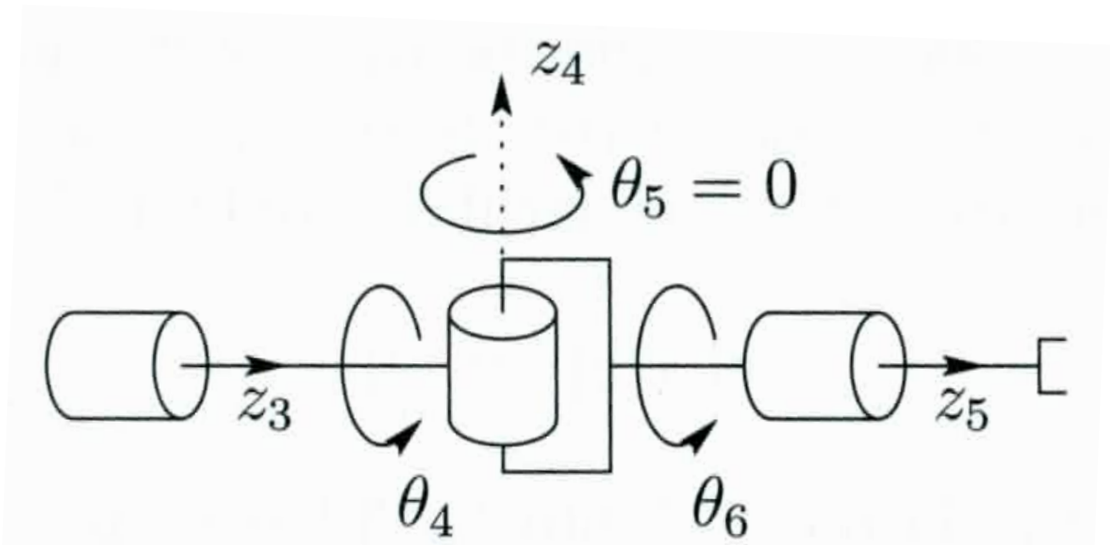
$$J = [J_{\text{arm}} \mid J_{\text{wrist}}] = \left[\frac{J_{11}}{J_{21}} \middle| \frac{J_{12}}{J_{22}}\right]$$

if we choose $o_4 = o_5 = o_6$

$$J = \left[\frac{J_{11}}{J_{21}} \middle| \frac{0}{J_{22}}\right]$$

$$\det(J) = \det(J_{11})\det(J_{22})$$

arm wrist



$\theta_5 = 0, \pi$ are singular configurations

The transpose of the Jacobian relates joint forces and torques to Cartesian end-effector forces and torques

(n × 1) (n × 6) (6 × 1)

$$\vec{\tau} = J^T(\vec{q})\vec{F}$$

joint forces and torques

Jacobian matrix transpose

endpoint forces and torques

Simplest to think about for a 3-DOF robot with all revolute joints.

We want to output a force at the tip.

(3 × 1) (3 × 3) (3 × 1)

$$\vec{\tau} = J^T(\vec{q})\vec{F}$$

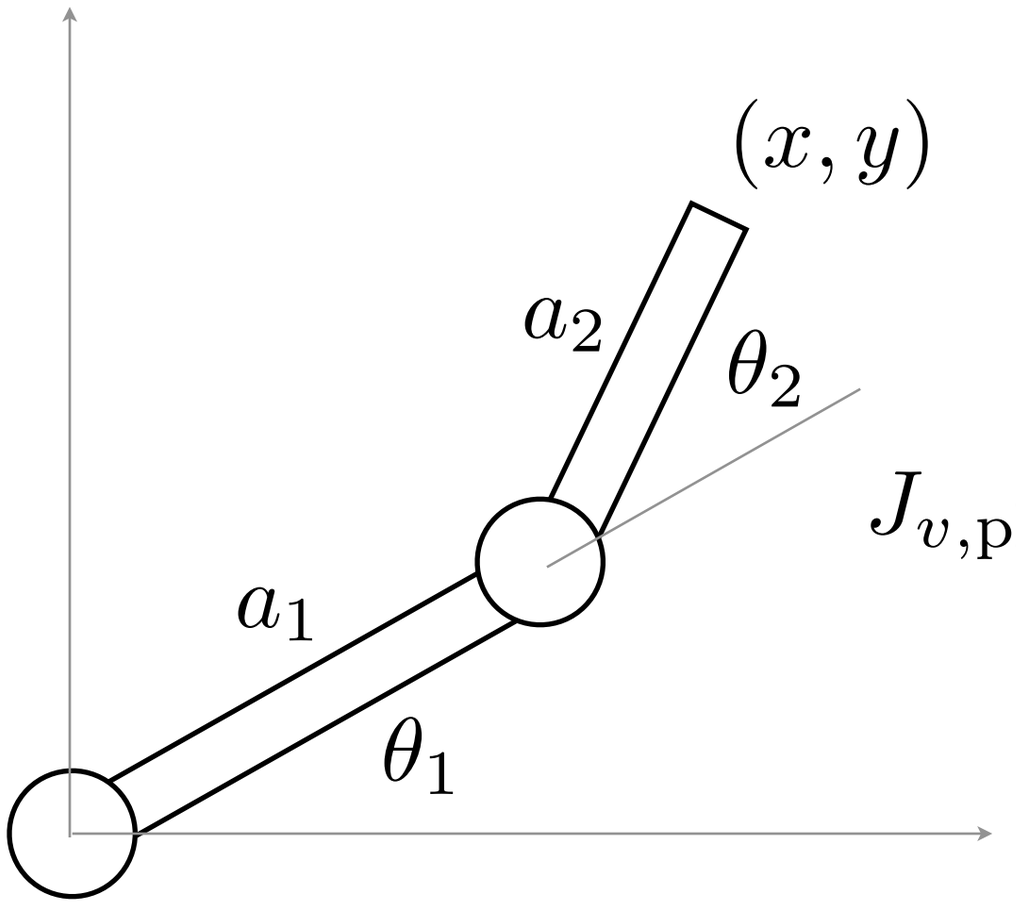
joint torques

Jacobian matrix transpose

endpoint forces

$$\vec{\tau} = J^T(\vec{q})\vec{F}$$

This relationship stems from virtual work.



Beginning with the 2 x 2 linear velocity Jacobian

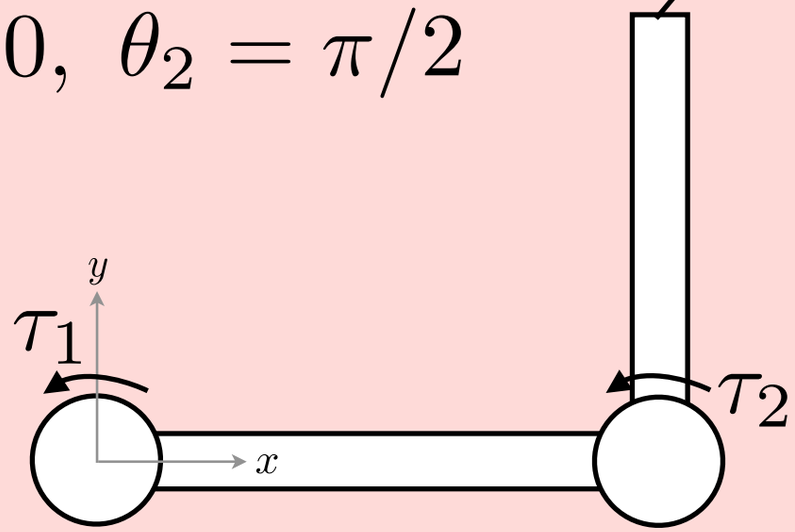
$$J_{v,\text{planar}}(\vec{q}) = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & -a_2 s_{12} \\ a_1 c_1 + a_2 c_{12} & a_2 c_{12} \end{bmatrix}$$

We can solve for the joint torques necessary to exert a desired force at the end effector using the Jacobian transpose

$$\vec{\tau} = J^T(\vec{q})\vec{F}$$

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & a_1 c_1 + a_2 c_{12} \\ -a_2 s_{12} & a_2 c_{12} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \end{bmatrix}$$

$$\theta_1 = 0,\ \theta_2 = \pi/2$$



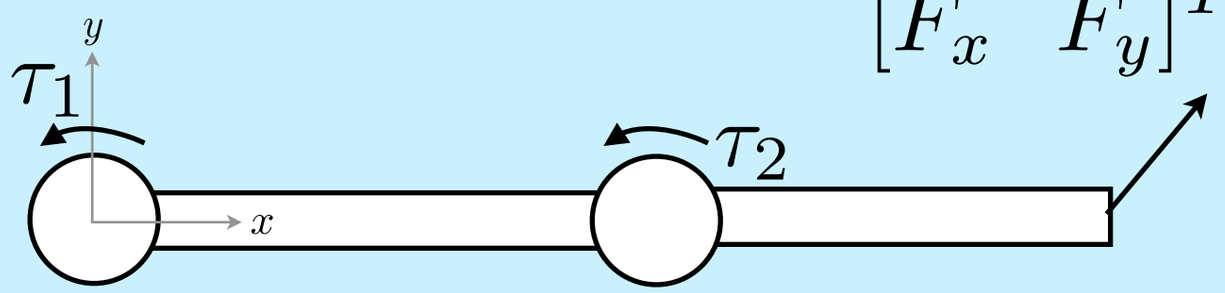
$$J_v([0 \quad \pi/2]^T) = \begin{bmatrix} -a_2 & -a_2 \\ a_1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\tau_1 = -a_2 F_x + a_1 F_y$$

$$\tau_2 = -a_2 F_x$$

Can create forces in both x and y directions.

$$\theta_1 = 0,\ \theta_2 = 0$$



$$J_v([0 \quad 0]^T) = \begin{bmatrix} 0 & 0 \\ a_1 + a_2 & a_2 \\ 0 & 0 \end{bmatrix}$$

$$\tau_1 = (a_1 + a_2) F_y$$

$$\tau_2 = a_2 F_y$$

Can't create forces in the x direction!

Questions ?

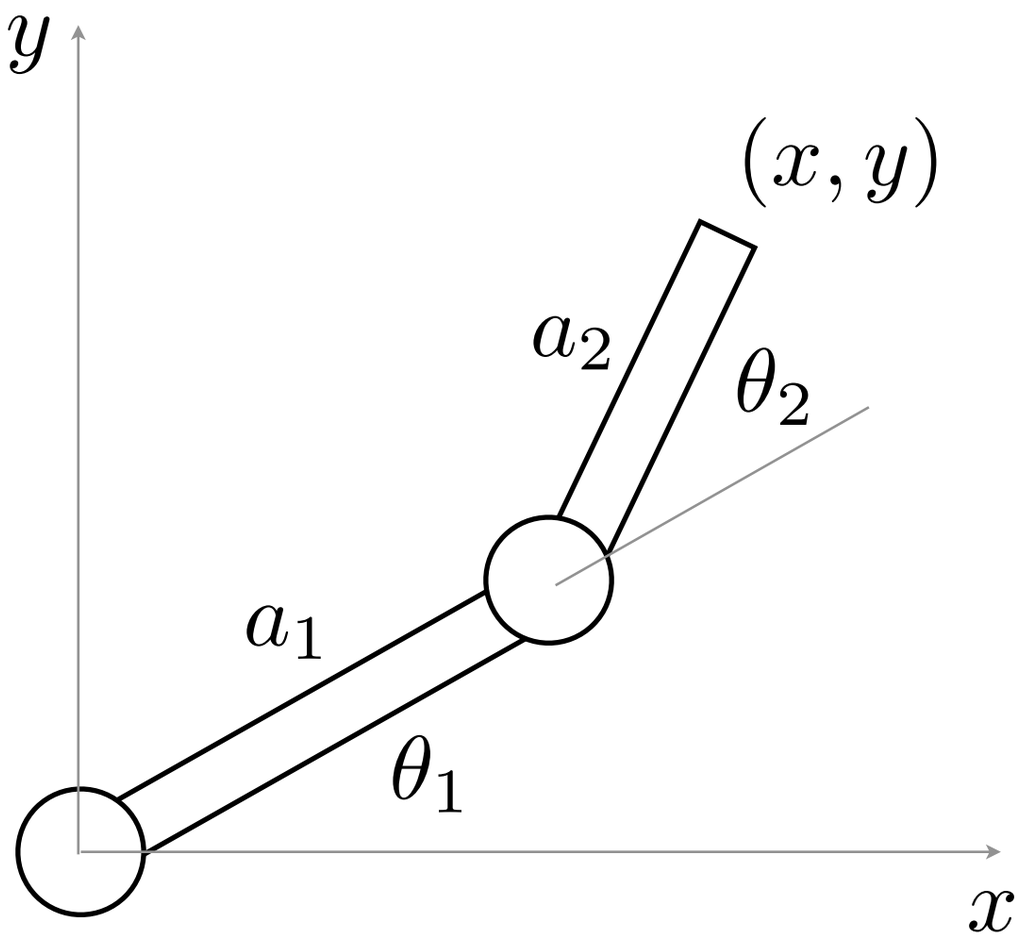
For a specific configuration, the Jacobian scales the input (joint velocities) to the output (body velocity)

$$\xi = J(q)\dot{q}$$

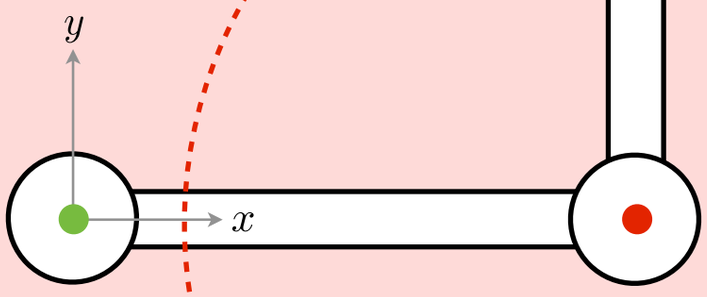
If you put in a joint velocity vector with unit norm, you can calculate in which direction and how fast the robot's end-effector will translate and rotate.

This approach allows you to calculate and plot the manipulability ellipsoid – a geometrical representation of all the possible tip velocities for a normalized joint velocity input.
A 6D ellipsoid is hard to visualize,
but 2D and 3D ellipsoids are lovely and useful.

What does the manipulability ellipsoid look like for the planar RR robot?



$\theta_1 = 0,\ \theta_2 = \pi/2$

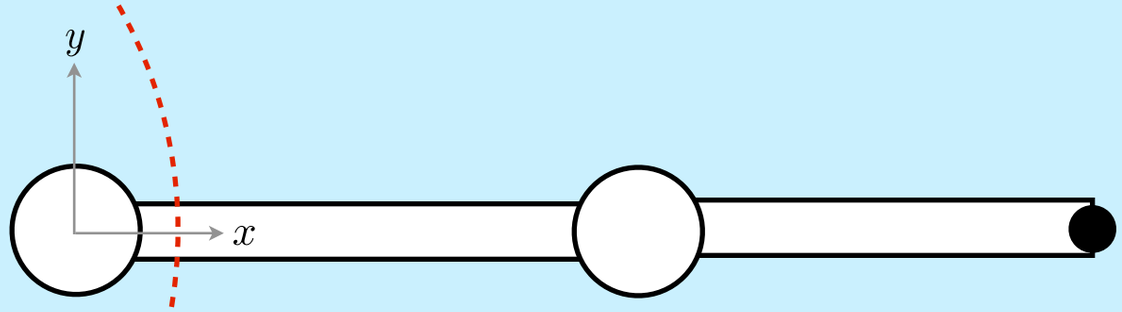


$$J_v([0 \quad \pi/2]^T) = \begin{bmatrix} -a_2 & -a_2 \\ a_1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -a_2\dot{\theta}_1 - a_2\dot{\theta}_2 \\ a_1\dot{\theta}_1 \\ 0 \end{bmatrix}$$

The robot's tip cannot move in the z direction, but it can move in both x and y directions...

$\theta_1 = 0,\ \theta_2 = 0$

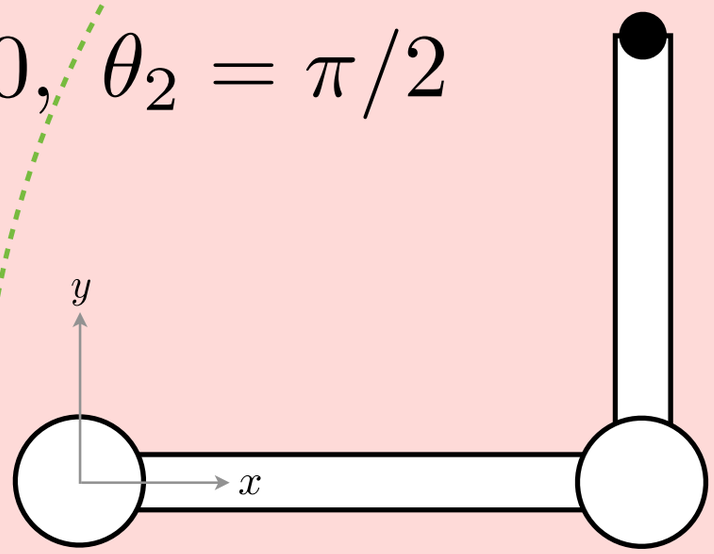


$$J_v([0 \quad 0]^T) = \begin{bmatrix} 0 & 0 \\ a_1 + a_2 & a_2 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ (a_1 + a_2)\dot{\theta}_1 + a_2\dot{\theta}_2 \\ 0 \end{bmatrix}$$

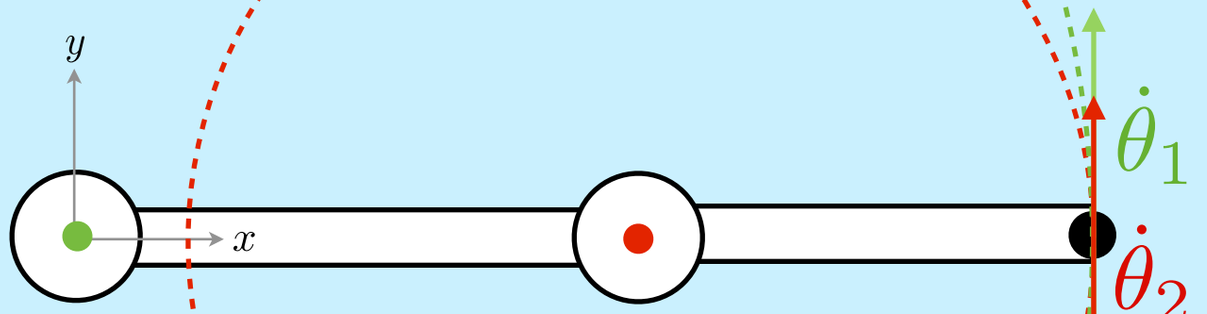
$$\dot{\vec{p}} = J_v(\vec{q})\,\dot{\vec{q}}$$

$$\theta_1 = 0,\ \theta_2 = \pi/2$$



$$J_v([0 \quad \pi/2]^T) = \begin{bmatrix} -a_2 & -a_2 \\ a_1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\theta_1 = 0,\ \theta_2 = 0$$



$$J_v([0 \quad 0]^T) = \begin{bmatrix} 0 & 0 \\ a_1 + a_2 & a_2 \\ 0 & 0 \end{bmatrix}$$

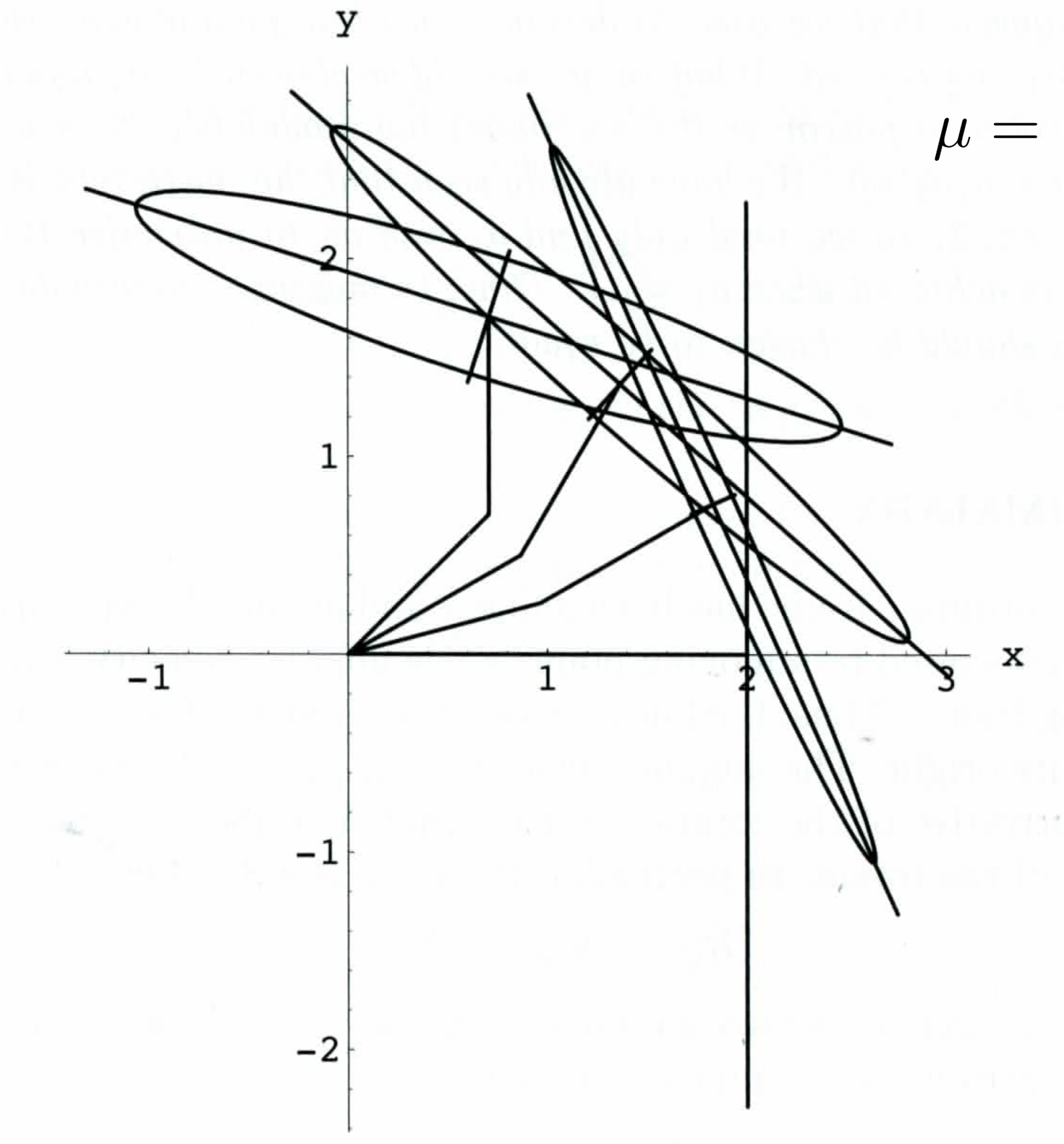
$$\dot{\vec{p}} = J_v(\vec{q})\,\dot{\vec{q}}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -a_2\dot{\theta}_1 - a_2\dot{\theta}_2 \\ a_1\dot{\theta}_1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ (a_1 + a_2)\dot{\theta}_1 + a_2\dot{\theta}_2 \\ 0 \end{bmatrix}$$

Notice anything else about this configuration?

The robot's tip cannot move in the x or z directions...



Manipulability

$$\mu = |\det(J)| = a_1 a_2 |\sin(\theta_2)|$$

Can be used to tell you where to perform certain tasks.

Also useful for deciding how to design a manipulator.

Questions ?

Homework 7:
PUMA 260 Singularities and Manipulability

MEAM 520, University of Pennsylvania
Katherine J. Kuchenbecker, Ph.D.

October 24, 2013

This assignment is due on **Sunday, November 3, by midnight (11:59:59 p.m.)** Your code should be submitted via email according to the instructions at the end of this document. Late submissions will be accepted until Wednesday, November 6, by midnight (11:59:59 p.m.), but they will be penalized by 10% for each partial or full day late, up to 30%. After the late deadline, no further assignments may be submitted.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you write down must be your own work, not copied from any other individual or team. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. If you get stuck, post a question on Piazza or go to office hours!

Individual vs. Pair Programming

You may do this assignment either individually or with a partner. If you do this homework with a partner, you may work with ***anyone except your partner from Project 1***. We want everyone in this class to gain experience working with a variety of partners. Consider using the "Search for Teammates!" tool on Piazza.

If you are in a pair, you should work closely with your partner throughout this assignment, following the paradigm of pair programming. You will turn in one MATLAB script for which you are both jointly responsible, and you will both receive the same grade. Please follow these pair programming guidelines, which were adapted from "All I really need to know about pair programming I learned in kindergarten," by Williams and Kessler, *Communications of the ACM*, May 2000:

- Start with a good attitude, setting aside any skepticism and expecting to jell with your partner.
- Don't start writing code alone. Arrange a meeting with your partner as soon as you can.
- Use just one computer, and sit side by side; a desktop computer with a large monitor is better for this than a laptop. Make sure both partners can see the screen.
- At each instant, one partner should be driving (using the mouse and keyboard or recording design ideas) while the other is continuously reviewing the work (thinking and making suggestions).
- Change driving/reviewing roles at least every thirty minutes, *even if one partner is much more experienced than the other*. You may want to set a timer to help you remember to switch.
- If you notice a bug in the code your partner is typing, wait until they finish the line to correct them.
- Stay focused and on-task the whole time you are working together.
- Recognize that pair programming usually takes more effort than programming alone, but it produces better code, deeper learning, and a more positive experience for the participants.
- Take a break periodically to refresh your perspective.
- Share responsibility for your project; avoid blaming either partner for challenges you run into.

Homework 7

MATLAB Assignment on PUMA 260 Singularities and Manipulability

Done individually or with a partner *(not project 1 partner)*

Due Sunday 11/3

Task Description

Like Homework 5 and Project 1, this assignment centers on the PUMA 260, an articulated (RRR) robot with lateral offsets plus a spherical wrist (RRR). As described in the steps below, your task is to write a MATLAB script that analyzes this robot's velocity kinematics, determines the singularities of its arm and wrist, and plots the ellipsoids that represent the manipulability of its arm and wrist for any pose specified by the user.

1. **Download and Rename Starter Code**
   Download the following three files from the Piazza course page under Resources/Homework:

   - `analyze_puma_starter.m` – A MATLAB script that you will update to do the requisite analysis of the PUMA's velocity kinematics.
   - `plot_puma_starter.m` – A MATLAB function that plots the PUMA robot along with an arbitrary sphere; you will update this function to include specified augmentations to the plot, such as changing the arbitrary sphere into a manipulability ellipsoid.
   - `dh_kuchenbe.m` – A MATLAB function that calculates the transformation matrix associated with the DH parameters `a`, `alpha`, `d`, and `theta`. Because the angles are specified in radians, this function can take both numerical and symbolic arguments. You do not need to modify this file.

   Rename the first two files with your PennKey (plus that of your partner, if you are working in a pair): for example, either `analyze_puma_pennkey.m` or `analyze_puma_pennkey1_pennkey2.m`.

2. **Update Function Name**
   Open both renamed files in MATLAB. Change the function declaration on the top line of the second file (`plot_puma_starter`) so that it matches your filename. In the first file, search for `_starter` and replace it with an underscore and your PennKey string so that all of the calls go to your new version of this function.

3. **Run Starter Code**
   Run your renamed version of `analyze_puma_starter.m` to be sure everything is set up correctly. Look at the code and what it outputs.

4. **Modify Puma Plot for Decoupling**
   It is mathematically challenging to calculate all of the linear and angular velocity singularities of a general 6-DOF manipulator because doing so requires one to analyze the determinant of a complicated six-by-six matrix. As discussed in class and detailed in SHV 4.9.1, you can simplify this problem by placing o_6, the origin of the sixth coordinate frame, at the wrist center instead of at the tip of the end-effector. Modify your `plot_puma` function in this way, including a comment to show what you did. (Note that the book erroneously states that you must choose the coordinate frames so that $o_3 = o_4 = o_5 = o_6$; all that is truly necessary is $o_4 = o_5 = o_6$).

5. **Calculate the Arm's Symbolic Linear Velocity Jacobian**
   MATLAB allows you to create symbolic variables using the keyword `syms`; for example, the command `syms th1 th2 th3 th4 th5 th6 real` creates six symbolic variables named `th1` through `th6` and constrains them to have only real (not complex) values. The provided `dh_kuchenbe(a, alpha, d, theta)` function was designed to be able to take both numerical and symbolic arguments.

   Defining the wrist center ($o_4 = o_5 = o_6$) as the end-effector position, calculate the symbolic 3×3 matrix that relates the arm's joint velocities ($\dot{\theta}_1,\ \dot{\theta}_2,\ \dot{\theta}_3$) to the end-effector's Cartesian velocities in the base frame ($\dot{x}_6^0,\ \dot{y}_6^0,\ \dot{z}_6^0$), as a function of the arm's current configuration (`th1` through `th6`). The MATLAB functions `diff` and/or `cross` may be useful to you. Use symbolic variables such as `a` to represent any constant linear DH parameters, but use actual angle values for any constant angular DH parameters such as `0` or `pi/2`. Call this symbolic matrix `Jv`.

Homework 7

MATLAB Assignment on PUMA 260 Singularities and Manipulability

Done individually or with a partner *(not project 1 partner)*

Due Sunday 11/3

Inspect the matrix that you calculated to determine whether it is correct, and fix any errors that you spot. One fruitful approach is to set the joint variables to specific values, such as the joint angles of the robot's home position, and calculate the numerical version of the Jacobian matrix by running the command `evalJv`, allowing the output to print to the command line so you can inspect it.

6. **Determine the Arm's Linear Velocity Singularities**
   Use your symbolic linear velocity Jacobian to calculate when the arm's singularities occur. The MATLAB functions `det`, `simple`, and `eval` may be useful to you. Display the simplified expression on the command line, and look at it until it makes sense (or until you realize you made a mistake).

7. **Plot the Arm's Linear Velocity Singularities**
   In the designated area of the script, choose a set of joint of angles that represent each of the PUMA arm's possible singularities, and plot the robot in each of these poses. Use the first figure window for the first type of singularity, the second for the second, etc. Even though you haven't yet augmented the plotting function to handle this, pass in the numerical 3×3 matrices for `Jv` as the seventh argument for the plot; you can calculate this from by setting appropriate symbolic variables to numbers and using the command `Jvnum = eval(Jv)`. Because this plot concerns linear velocities, pass in a 3×3 matrix of zeros for `Jw`. Also display a brief explanation of the geometric meaning of each of these singularities on the command line.

8. **Calculate the Wrist's Symbolic Angular Velocity Jacobian**
   Calculate the symbolic 3×3 matrix that relates the wrist's joint velocities ($\dot{\theta}_4,\ \dot{\theta}_5,\ \dot{\theta}_6$) to the end-effector's angular velocity relative to the base frame in the base frame ($\vec{\omega}^0_{0,6}$), as a function of the arm's current configuration (`th1` through `th6`). Call this symbolic matrix `Jw`.

9. **Determine the Wrist's Angular Velocity Singularities**
   Use your symbolic angular velocity Jacobian to calculate when the wrist's singularities occur. The MATLAB functions `det`, `simple`, and `eval` may be useful to you. Display the simplified expression on the command line, and look at it until it makes sense (or until you realize you made a mistake).

10. **Plot the Wrist's Angular Velocity Singularities**
    In the designated area of the script, choose a set of joint of angles that represent each of the PUMA wrist's possible singularities, and plot the robot in each of these poses. Use the next figure window for the first type of singularity, etc. Even though you haven't yet augmented the plotting function to handle this, pass in the numerical 3×3 matrices for `Jw` as the eigth argument for the plot. Because this plot concerns angular velocities, pass in a 3×3 matrix of zeros for `Jv`. Also display a brief explanation of the geometric meaning of each of these singularities on the command line.

11. **Plot the Robot in Another Interesting Configuration**
    Create one final plot of the robot in another figure window, setting the joint variables to arbitrary values. Pass in the correct `Jv` and `Jw` matrices or zero matrices, as desired. Use this plot to debug the vector and ellipsoid plotting of the next steps and to deepen your understanding of this material.

12. **Plot Linear Velocity Vectors**
    Update your plotting function (`plot_puma_pennkey`) to graph three velocity vectors as warm-colored line segments emanating from the end-effector. Use a scale of 1 in. per in./s.

    - The first line segment should be magenta and show the linear velocity vector the tip would experience if only joint 1 was activated with unit velocity (+1 rad/s).
    - The second line segment should be red and show the linear velocity vector the tip would experience if only joint 2 was activated with unit velocity (+1 rad/s).
    - The third line segment should be yellow and show the linear velocity vector the tip would experience if only joint 3 was activated with unit velocity (+1 rad/s).

Homework 7

MATLAB Assignment on PUMA 260 Singularities and Manipulability

Done individually or with a partner
(not project 1 partner)

Due Sunday 11/3

13. **Plot Linear Velocity Manipulability Ellipsoid**
Update your plotting function (`plot_puma_pennkey`) to graph the robot's linear velocity manipulability ellipsoid as a transparent warm-colored ellipsoid centered at the end-effector. As explained somewhat cryptically in SHV 4.12, the manipulability ellipsoid shows how far and in what direction the robot's end-effector would move for a joint velocity vector $\dot{\vec{q}}$ that has unit norm, i.e., $\dot{\vec{q}} \cdot \dot{\vec{q}} = \dot{\vec{q}}^T \dot{\vec{q}} = 1$. For this plot, consider only motion of the robot's first three joints (the arm), corresponding to the symbolic `Jv` that you calculated above and that is being passed in numerically to the plotting function. Graph your ellipsoid at a scale of 0.5 in. per 1.0 in./s (half the scale of the velocity vectors) so you can see it well.

    To help you with this plotting task, the starter code plots a scaled transparent colored sphere at an arbitrary location in the PUMA's workspace. (I bet you were wondering why that was there!) Use the plots of the robot that you have created to test this plotting function – what should the linear velocity manipulability ellipsoid look like when the robot is at a linear velocity singularity?

14. **Plot Angular Velocity Vectors**
Update your plotting function (`plot_puma_pennkey`) to graph three angular velocity vectors as cool-colored line segments emanating from the end-effector. Use a scale of 10 in. per rad/s.

    - The first line segment should be green and show the angular velocity vector the tip would experience if only joint 4 was activated with unit velocity (+1 rad/s).
    - The second line segment should be cyan and show the angular velocity vector the tip would experience if only joint 5 was activated with unit velocity (+1 rad/s).
    - The third line segment should be blue and show the angular velocity vector the tip would experience if only joint 6 was activated with unit velocity (+1 rad/s).

15. **Plot Angular Velocity Manipulability Ellipsoid**
Update your plotting function (`plot_puma_pennkey`) to graph the robot's angular velocity manipulability ellipsoid as a transparent cool-colored ellipsoid centered at the end-effector. For this plot, consider only motion of the robot's last three joints (the wrist), corresponding to the symbolic `Jw` that you calculated above and that is being passed in numerically to the plotting function. Make this ellipsoid have cool (not warm) colors so it looks different from the linear velocity ellipsoid. It is fine to assume that your function will need to plot only one of the two ellipsoids at a time, with the other being zeros. Use a scale of 5 in. per rad/s (half the scale of the angular velocity vectors) so you can see it well.

    Use the plots of the robot that you have created to test this plotting function – what should the angular velocity manipulability ellipsoid look like when the robot is at an angular velocity singularity?

Submitting Your Code

Follow these instructions to submit your code:

1. Start an email to `meam520@seas.upenn.edu`
2. Make the subject *Homework 7: Your Name* or *Homework 7: Your Name and Your Teammate's Name*, replacing *Your Name* and *Your Teammate's Name* with the appropriate full names.
3. Attach your correctly named MATLAB files (`analyze_puma_pennkey.m` and `plot_puma_pennkey.m` plus any other functions you created, similarly identified) to the email. Please do not put them in a zip file or include any other attachments.
4. Optionally include any comments you have about this assignment and the experience of pair programming if you worked with a teammate.
5. Send the email.

You are welcome to resubmit your code if you want to make corrections. To avoid confusion, please state in the new email that it is a resubmission, and include all of your MATLAB files, even if you have not updated all of them.

Homework 7

MATLAB Assignment on PUMA 260 Singularities and Manipulability

Done individually or with a partner
(not project 1 partner)

Due Sunday 11/3

Editor - /Users/kuchenbe/Documents/teaching/meam 520/assignments/07 manipulability/matlab/analyze_puma_starter.m

EDITOR | PUBLISH | VIEW

analyze_puma_starter.m | plot_puma_starter.m | dh_kuchenbe.m

```
1   %% analyze_puma_starter.m
2   %
3   % This Matlab script is part of the starter code for Homework 7 in MEAM 520
4   % at the University of Pennsylvania.
5
6
7   %% SETUP
8
9   % Clear all variables from the workspace.
10  clear all
11
12  % Home the console, so we can more easily find any errors that may occur.
13  home
14
15  % Set student names.
16  studentNames = 'PUT YOUR NAME HERE';
17
18
19  %% CALCULATE THE ARM'S SYMBOLIC LINEAR VELOCITY JACOBIAN
20
21  % Define real-valued symbolic variables for all six joint angles.
22  syms th1 th2 th3 th4 th5 th6 real
23
24  % Put your calculations here.
25
26  % For now, set Jv equal to a 3x3 zeros matrix.  You must change this.
27  Jv = zeros(3,3);
28
29
30  %% DETERMINE THE ARM'S LINEAR VELOCITY SINGULARITIES
31
32  % Put your calculations here.
33
34  % Display the simplified symbolic determinant of Jv on the command line.
35  % For now I am just setting this to be equal to th1 for demo purposes.
36  detJv = th1
37
38
39  %% PLOT THE ARM IN EACH ARM SINGULARITY
40
41  % Explain when the first singularity occurs.
42  disp('As shown in Figure 1, the PUMA arm''s first singularity occurs when...')
43
44  % Set the values of the joint angles for the first singularity.  Here I am
45  % just using the home configuration for demo purposes.
46  theta1 = 0;
47  theta2 = 0;
48  theta3 = 0;
49  theta4 = 0;
50  theta5 = -pi/2;
```

script | Ln 5 Col 1

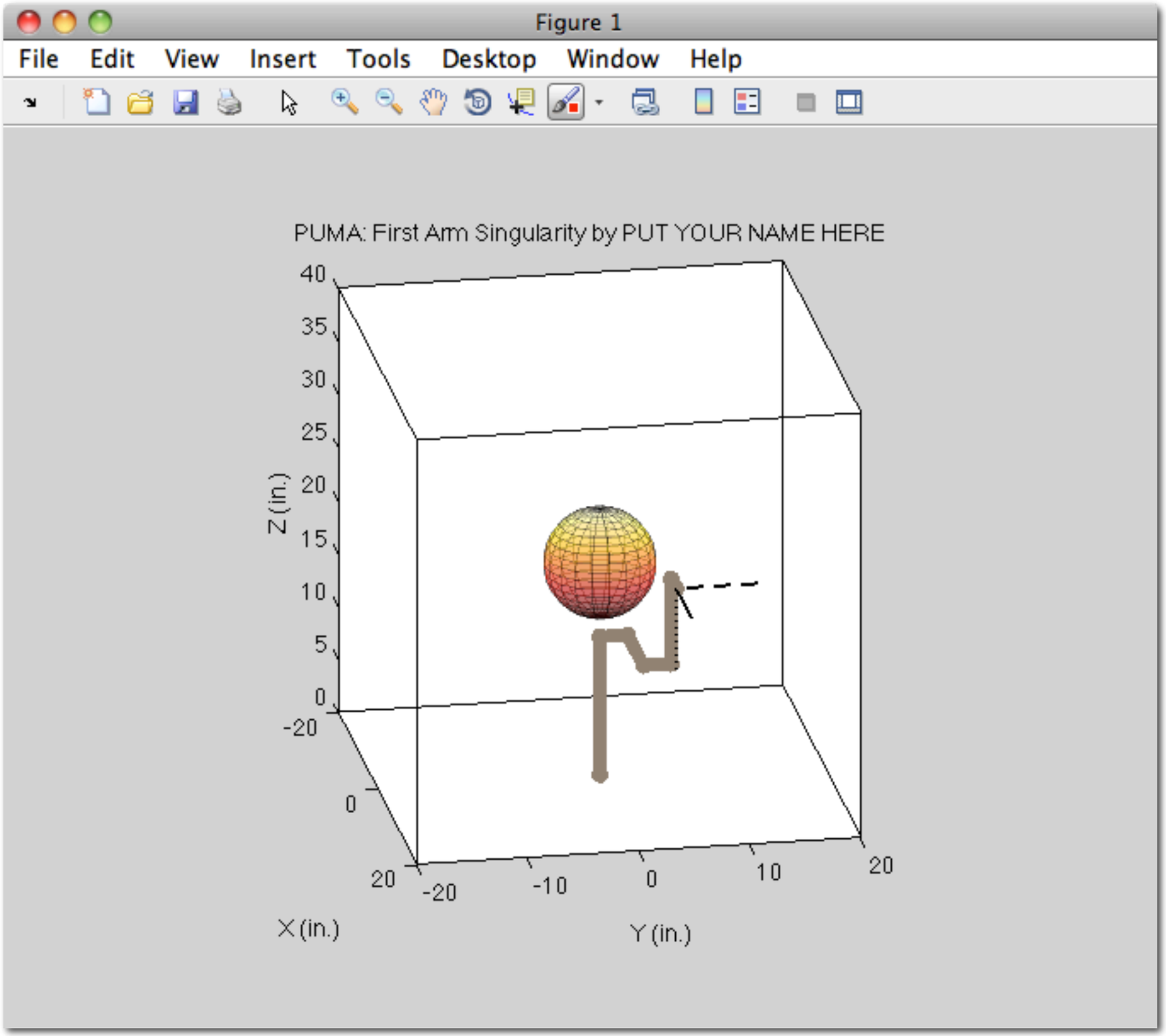


Figure 1
File Edit View Insert Tools Desktop Window Help
PUMA: First Arm Singularity by PUT YOUR NAME HERE
Z (in.)
40
35
30
25
20
15
10
5
0
-20
0
20
X (in.)
-20
-10
0
10
20
Y (in.)

Unimation

PUMA 260

Slides created by
Jonathan Fiene

May 1984
New Information
Mailed To: E, C, D/22-505A, C, E

A compact, computer-controlled robot for high speed, close-tolerance assembly, light materials handling, and inspection applications.

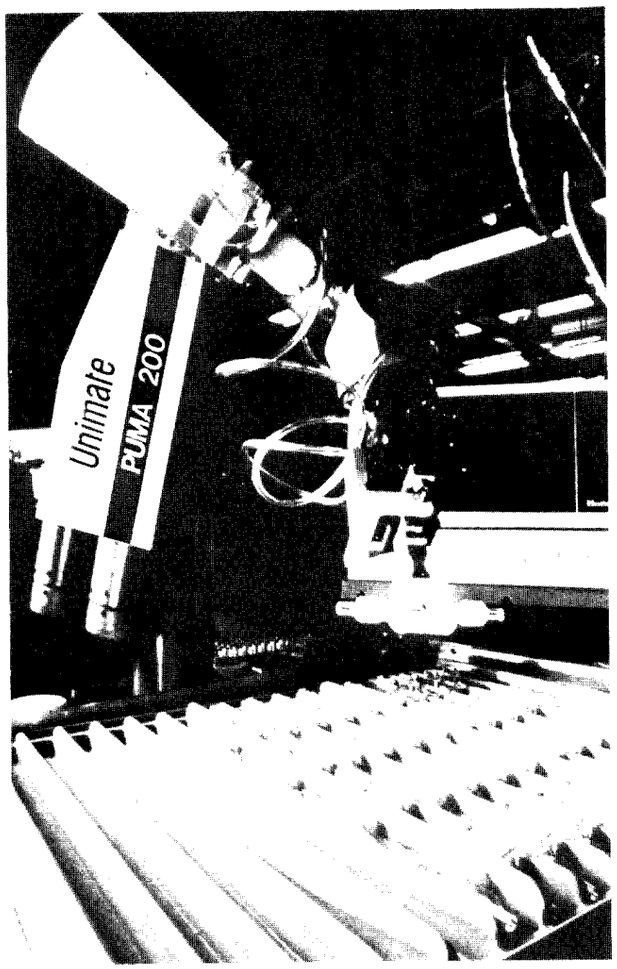
UNIMATE® PUMA® Series 200 Industrial Robot

FEATURES

The Series 200 is the most compact model in the UNIMATE PUMA line of electrically driven industrial robots. With an 18-inch reach and 2.2-pound payload capacity, the PUMA Series 200 robot is designed for medium to high-speed assembly and materials handling applications. Its capabilities are particularly suited to the requirements of electronics and other industries where lightweight parts handling is highly repetitive, fast and precise.

EASE OF USE

VAL™ a revolutionary advance in robot control systems, is used to control and program PUMA robots. The system uses an LSI-11 as a central processing unit and communicates with individual joint processors for servo control of robot arm motions. The results are ease in set up, high-tolerance repeatability, and greater application versatility.



VAL combines a sophisticated, easy to use robot programming capability with advanced servo control methods. Intuitive English language instruction provides fast, efficient program generation and editing capabilities. All servo-path computations are performed in real time, which makes it possible to interface with sensory-based systems.

EASE OF INSTALLATION

PUMA 200 robots are easily integrated into existing production lines because of the ease with which they are programmed. In addition, the robot can be easily and quickly reprogrammed for changes in the product or production process.

Programs can be written either on- or off-line using a CRT or teletype terminal, or they can be generated manually by guiding the robot arm through program paths using a microprocessor-based teach pendant. With either method, position data can be added or changed by key input, manual control or floppy disk input without affecting the overall task program.

VAL reduces memory requirements and permits complex programs, such as palletizing routines, to be easily written with a minimum number of taught positions.

Significant time savings can also be realized by integrating predefined subroutines and tasks into complex operations. These tasks can also be stored on floppy disks to build a library of routines, and even whole programs, that can be loaded into the memory of any PUMA controller so that specific tasks and repetitive routines need to be written only once.

With VAL, the Series 200 can also respond to fluctuations in the rate or other parameters of on-going production processes. Since all servo-path computations are performed in real time, changes in the arm path, or even task sequencing, can be initiated by feedback from various sensors and vision systems. As a result, the 200 can interact flexibly and efficiently as part of a large and complex manufacturing system.

APPLICATIONS

With its high speed, repeatability, and flexibility the PUMA 200 robot is suited to a wide range of small parts-handling applications, and VAL control makes it easy to design application programs to carry out the most difficult robotic tasks.

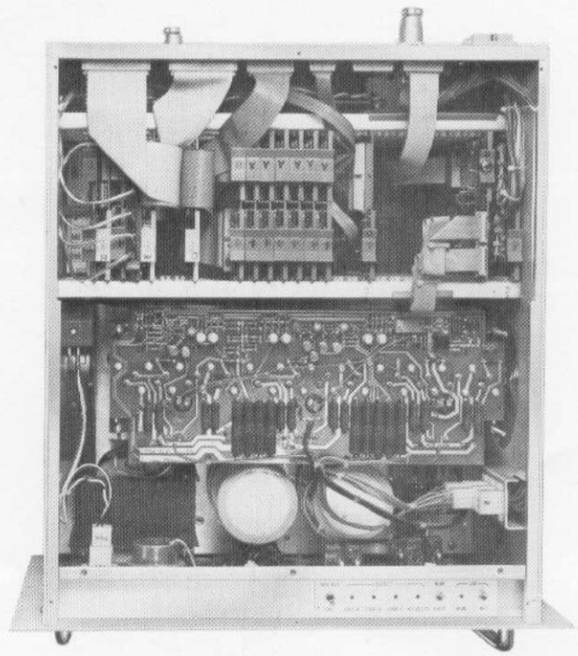
Current assembly applications include automotive instrument panels, small electric motors, printed circuit boards, subassemblies for radios, television sets, appliances and more. Other applications include packaging functions in the pharmaceutical, personal care, and food industries. Palletizing of small parts, inspection, and electronic parts handling in the computer, aerospace and defense industries round out the present installed base.

ADVANCED ENGINEERING

Modular, straightforward layout facilitating easy board replacement and plug-in expansion.

High-volume ventilation system.

Digital servo components designed for high-temperature operation and state-of-art dc motor control.



Compact packaging (19 in. rack mountable.).

Power amplifiers designed for high energy output with sensors for high temperature.

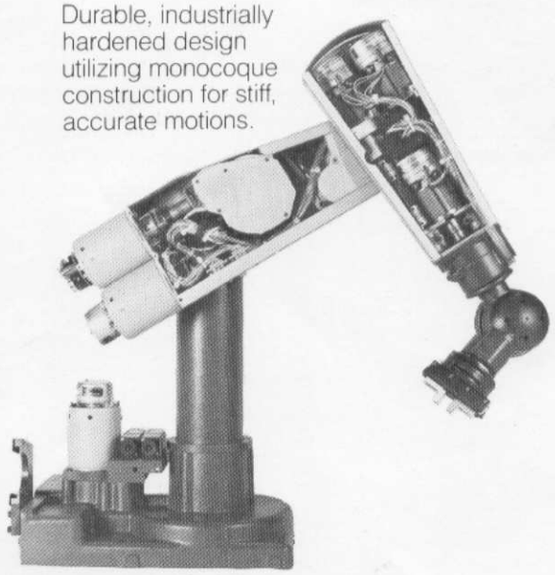
Auto-start button for automatic operation.

Easy-access panel location for self-diagnostic indicators and troubleshooting switches.

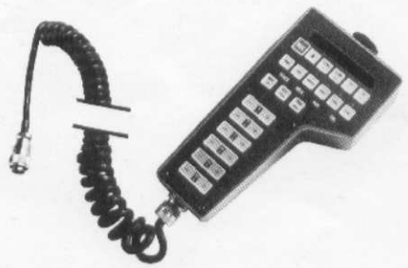
High-density, double-sided floppy-disk drive unit for program storage at 9,600 baud with 10,000-hour MTBF componentry.

High-precision gearing designed for ultra-low backlash and specially hardened for long life.

Durable, industrially hardened design utilizing monocoque construction for stiff, accurate motions.



Servo motors incorporating square wave encoders offer the most proven and sophisticated drive systems.



Industrial-grade, membrane-type teach pendant and CRT.

Unimate®

PUMA® Mark II Robot

200 Series **Equipment Manual** for VAL™ II and VAL™ PLUS Operating Systems

398V1

Unimation

A Westinghouse Company

Unimation Incorporated
Shelter Rock Lane
Danbury, Connecticut 06810

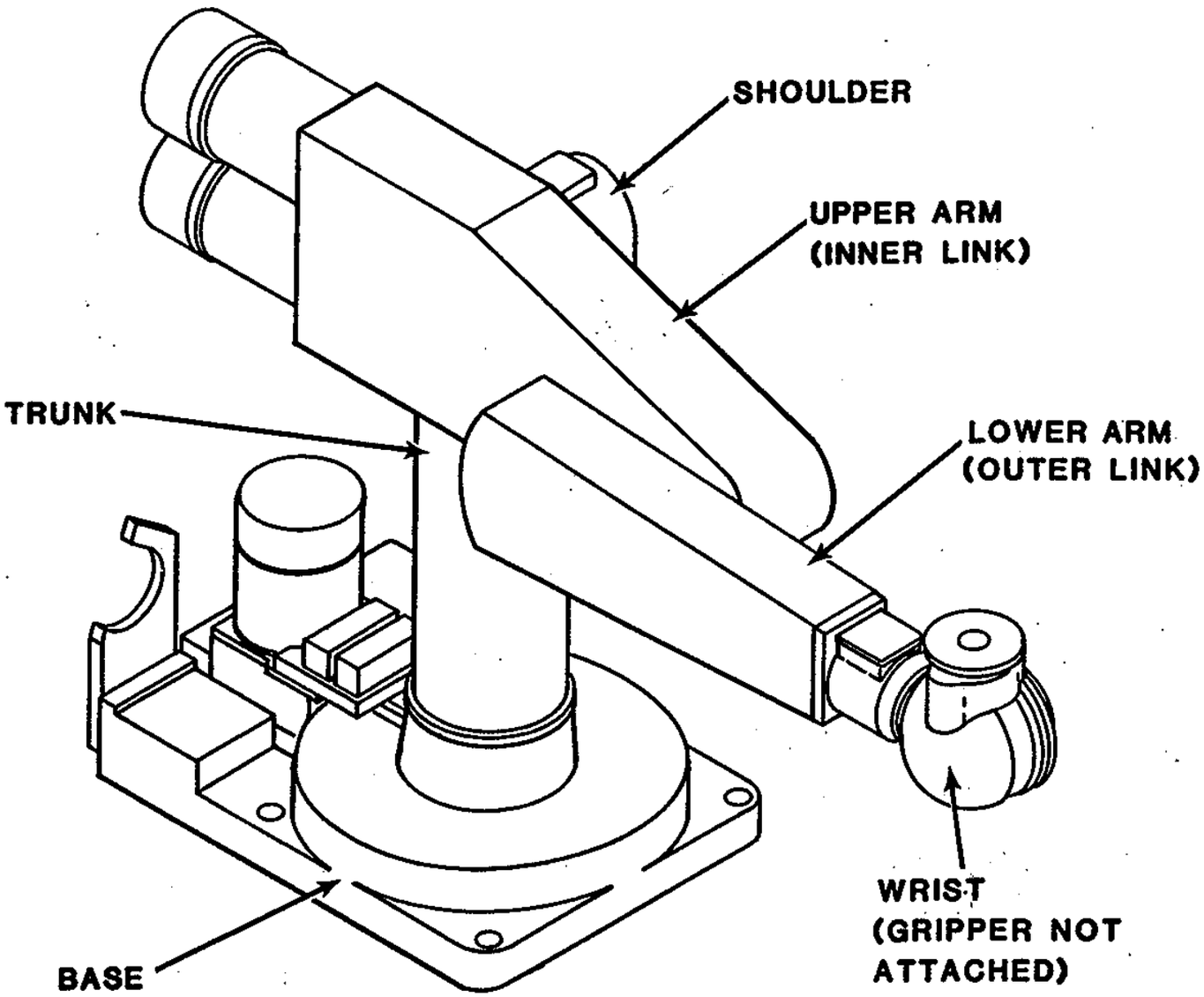


Figure 1-14. A Typical Video Terminal: CRT

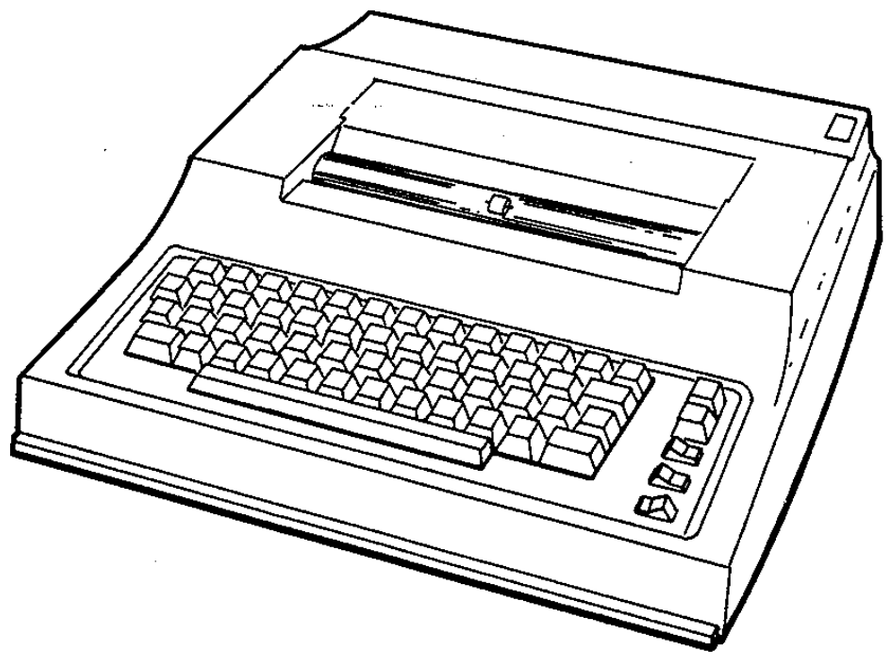


Figure 1-15. A Typical Hardcopy Terminal: TTY

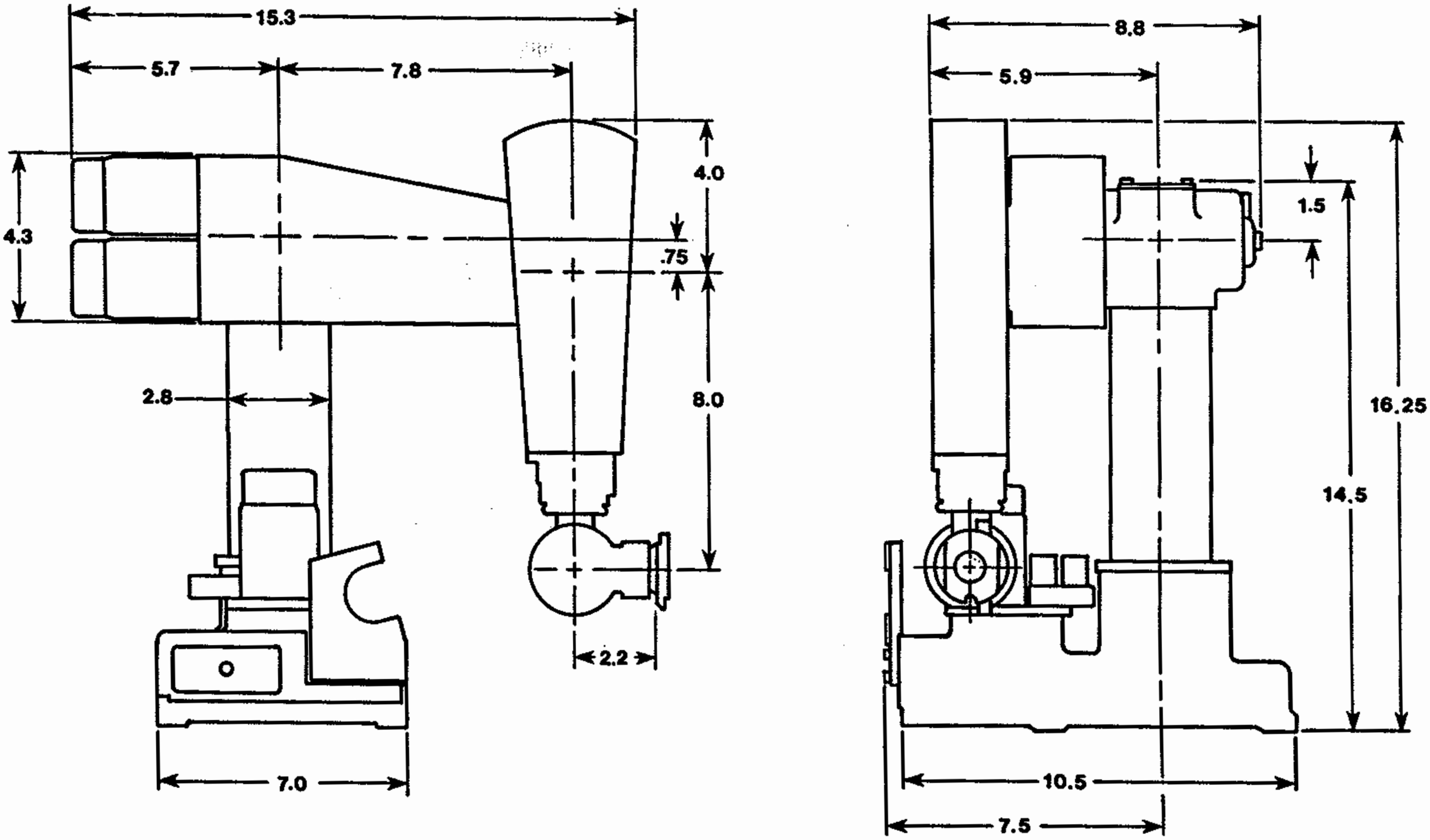
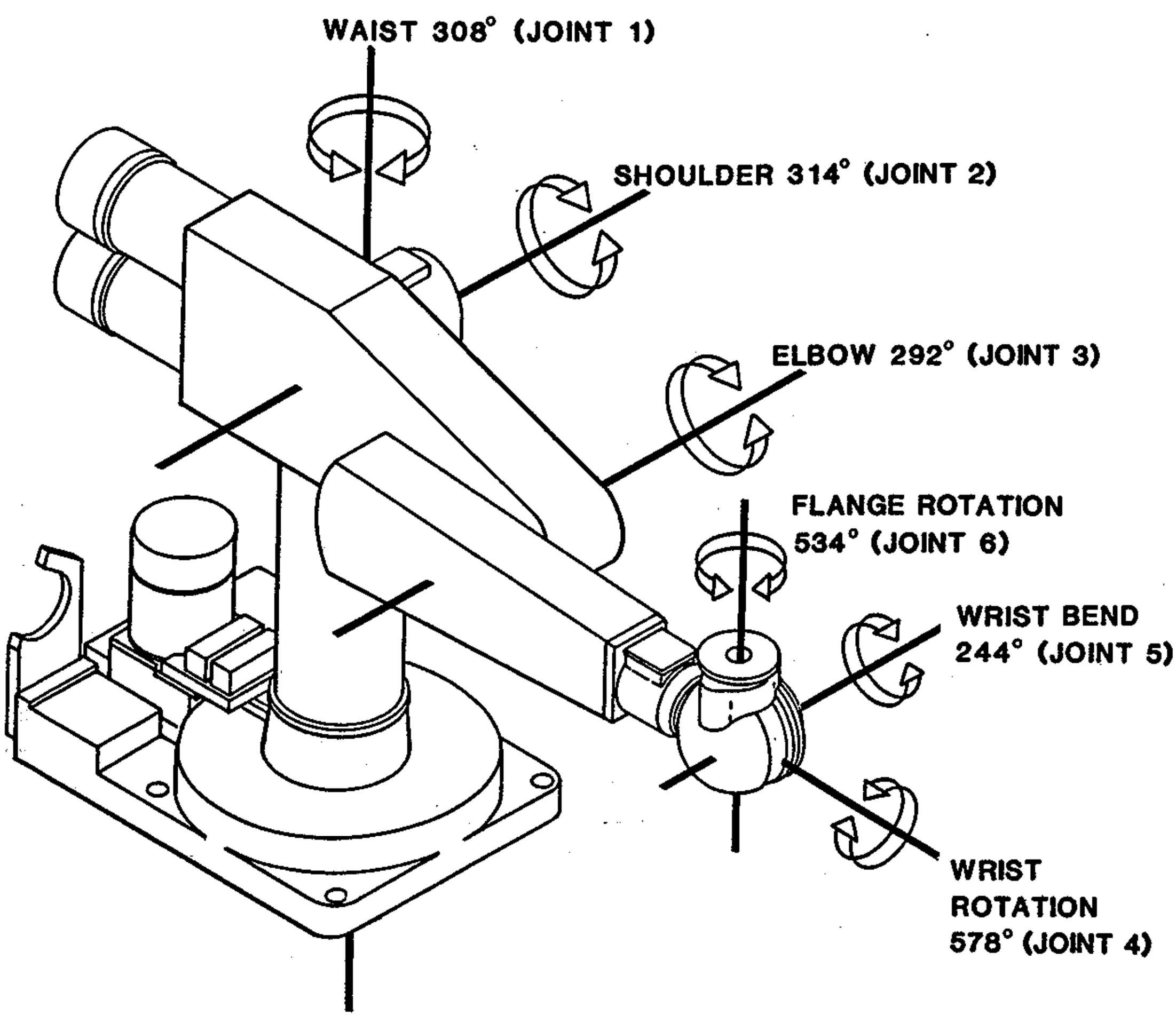
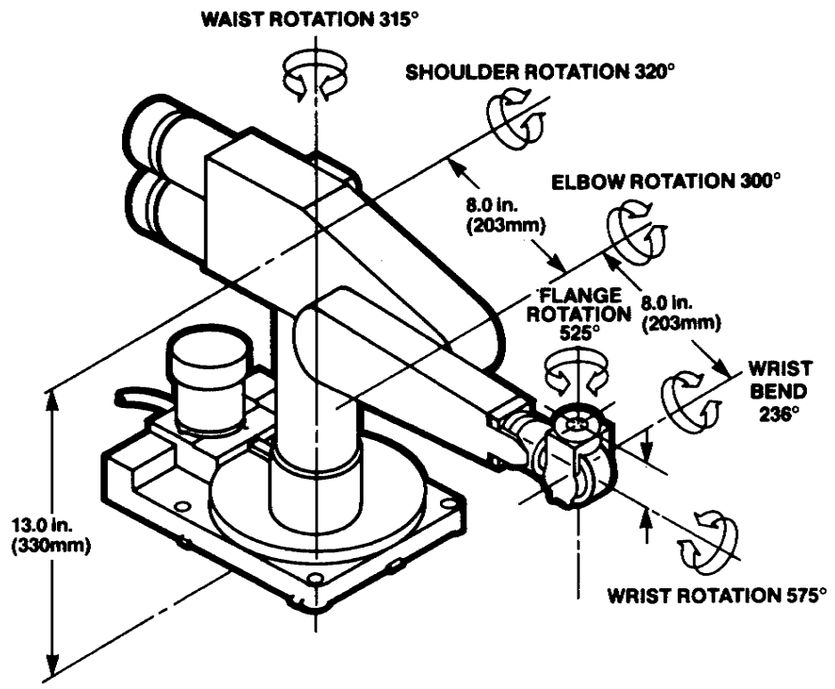


Figure 2-1. PUMA Dimensions - Installation


WAIST 308° (JOINT 1)
SHOULDER 314° (JOINT 2)
ELBOW 292° (JOINT 3)
FLANGE ROTATION
534° (JOINT 6)
WRIST BEND
244° (JOINT 5)
WRIST
ROTATION
578° (JOINT 4)

UNIMATE PUMA
200 Series



Performance

REPEATABILITY	±.002 in. (±0.05 mm)
LOAD CAPACITY	2.2 lbs. (1.0 Kg)
STRAIGHT LINE VELOCITY	49.0 in/s max. (1.25 m/s max.)
ENVIRONMENTAL REQUIREMENTS	50-120°F (10-50°C) 80% humidity (non-condensing). Shielded against industrial line fluctuations and human electro-static discharge

Physical Characteristics

ARM WEIGHT	15 lbs. (6.8 Kg)
CONTROLLER SIZE	19" x 12.5" x 23.6" (475 mm x 312.5 mm x 590 mm)
CONTROLLER WEIGHT	80 lbs. (36 Kg)
CONTROLLER CABLE LENGTH	15 ft. (4.5 m) Standard 50 ft. (15 m) Optional

General Specification

CONFIGURATION	6 revolute axes
DRIVE	Electric DC servo
CONTROLLER	System Computer (LSI-11/2 or 11/23)
Teaching Method	By manual control and/or computer terminal
Program Language	VAL or VAL II
Program Capacity	16K CMOS user memory std. (32K for VAL II)
External Program Storage	Floppy-disk (optional)
GRIPPER CONTROL	4-way pneumatic solenoid
POWER REQUIREMENT	110-130 V AC 50-60 Hz, 500 W
OPTIONAL ACCESSORIES	CRT or TTY terminals, floppy-disk memory storage, I/O module, 8 input/8 output signals (max. 32)—isolated AC/DC levels. Pneumatic gripper w/o fingers

What does the Puma's workspace look like?

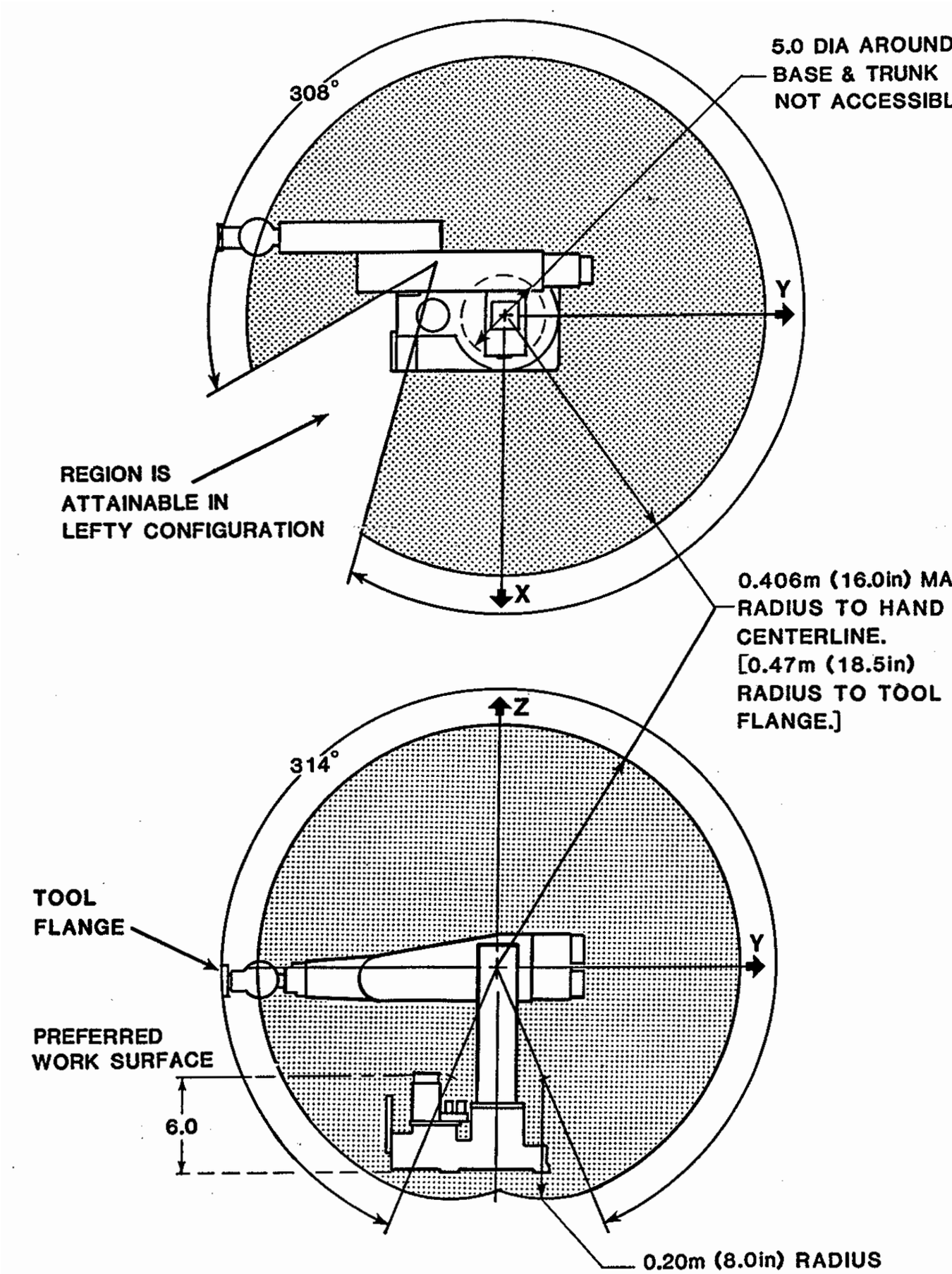
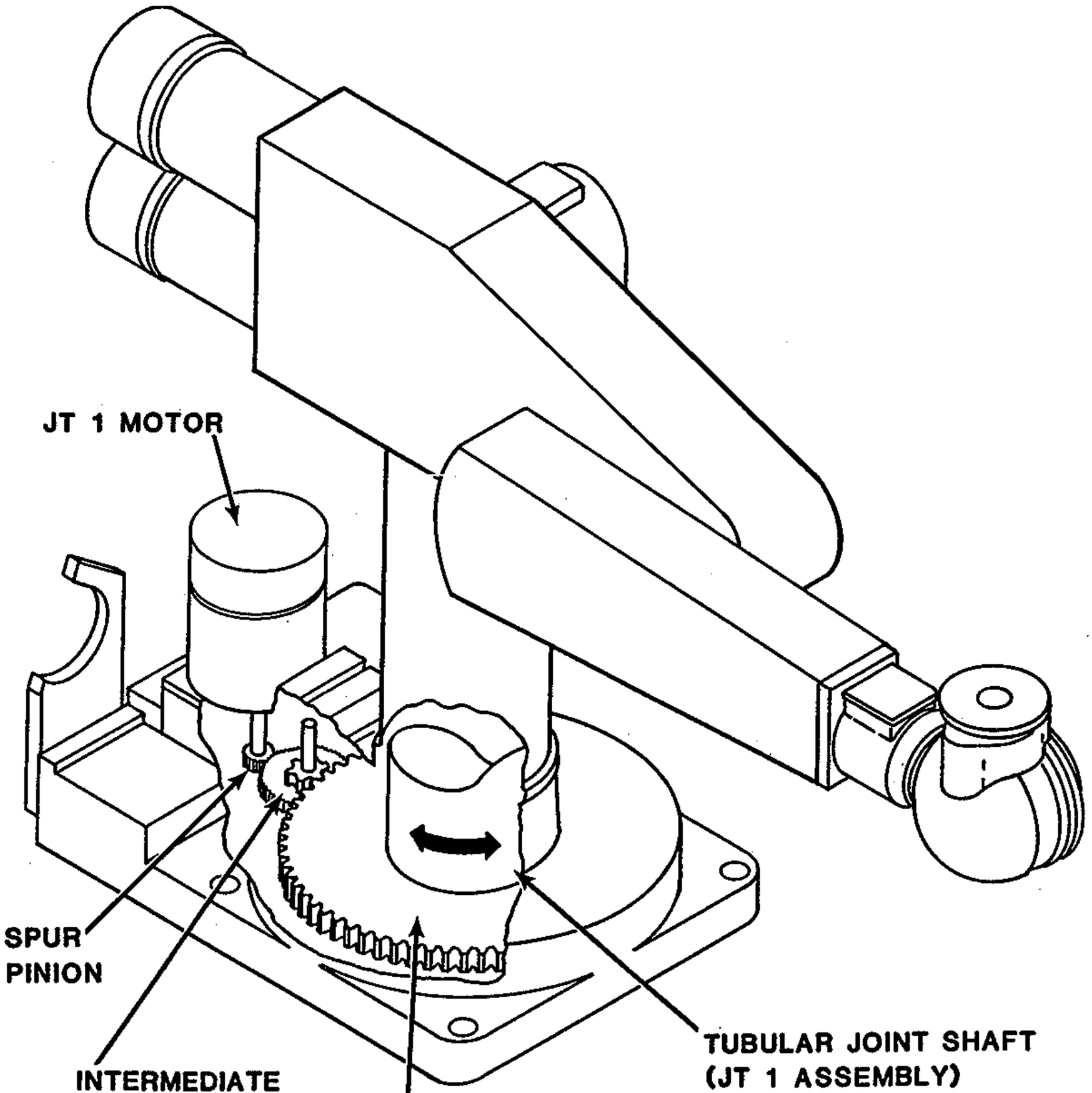
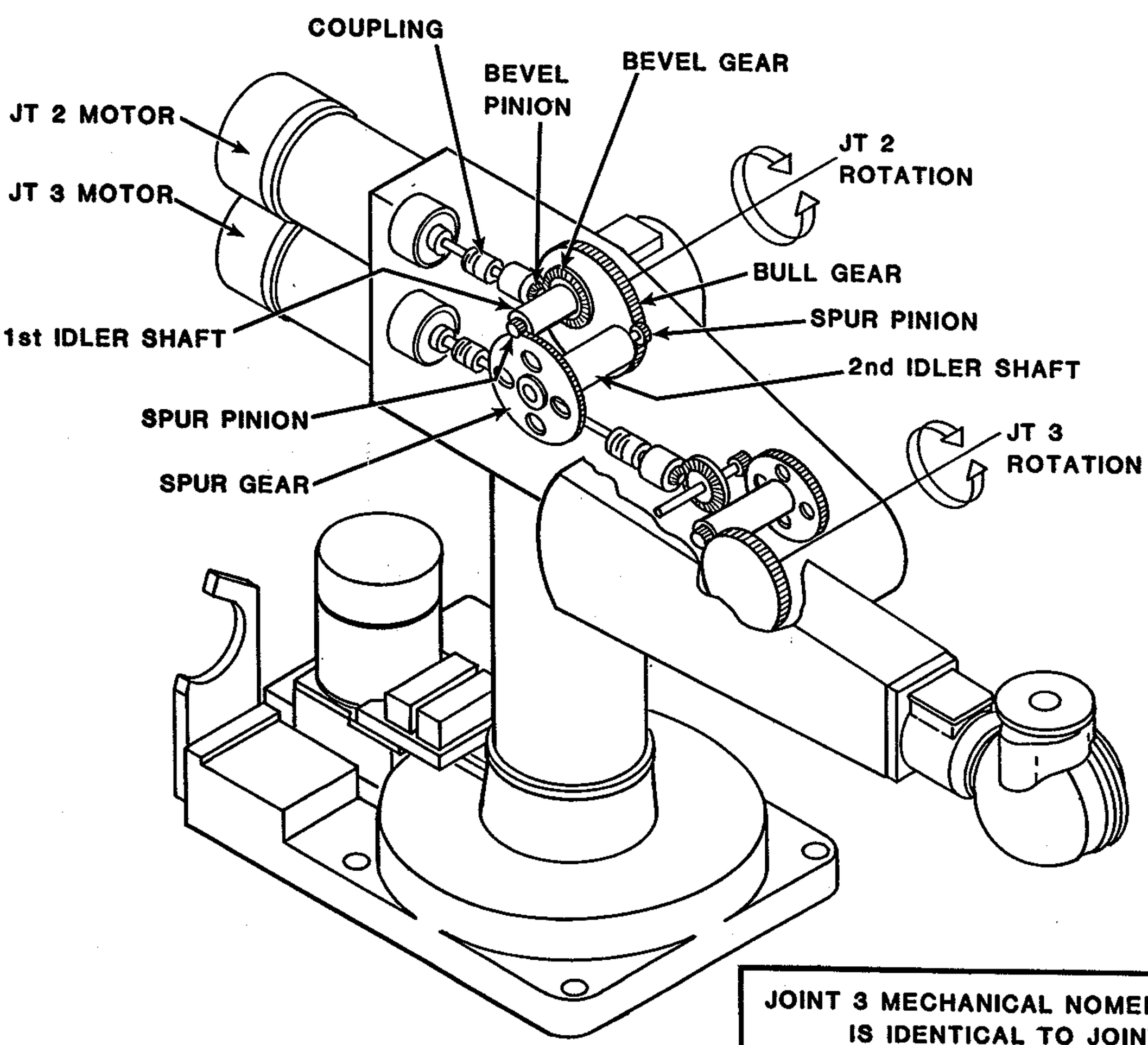


Figure 2-2. Robot Arm: Operating Envelope

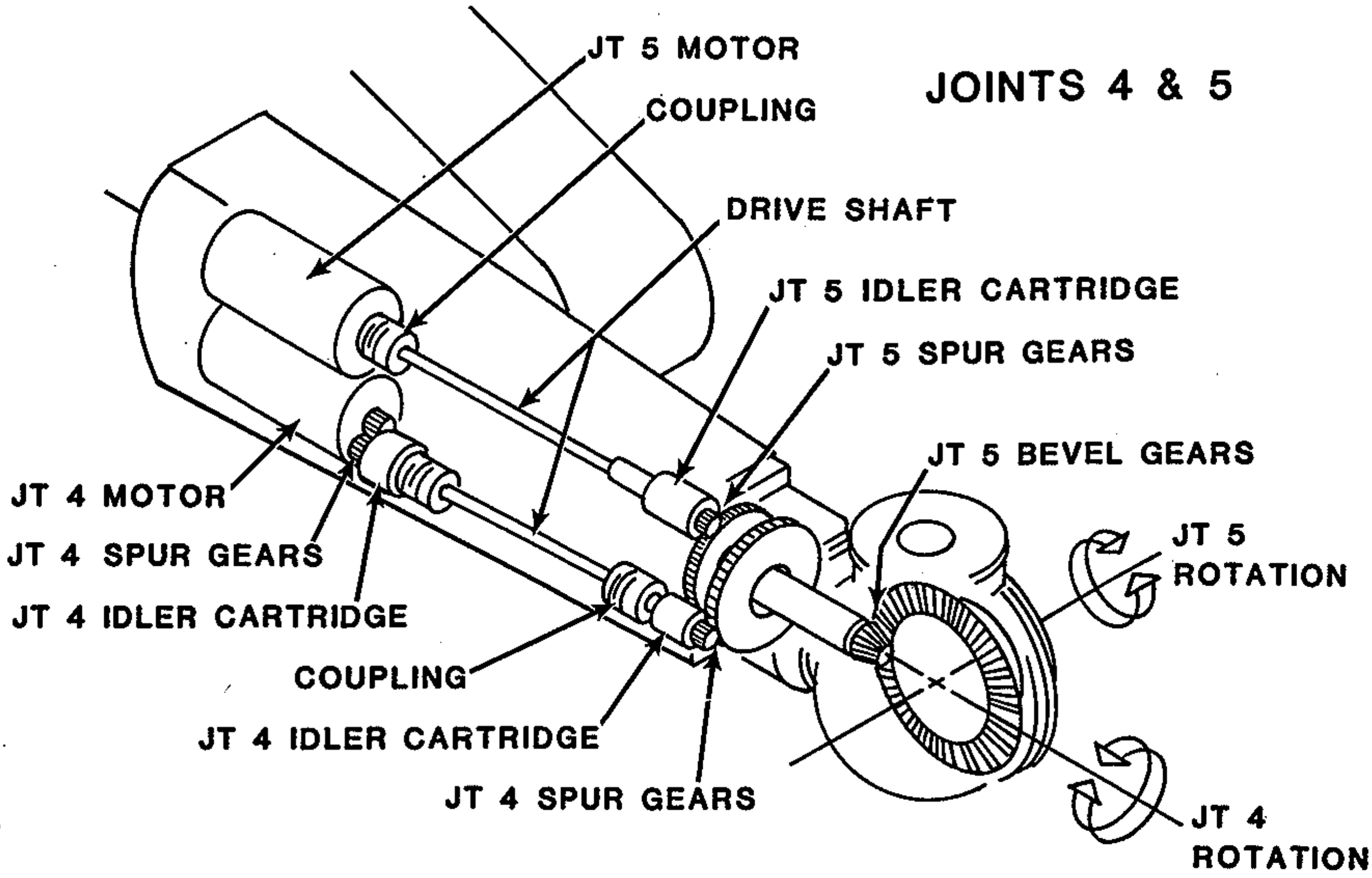
How does the Puma work?


JT 1 MOTOR
SPUR
PINION
INTERMEDIATE
IDLER GEARS
MAIN DRIVE
(BULL) GEAR
TUBULAR JOINT SHAFT
(JT 1 ASSEMBLY)

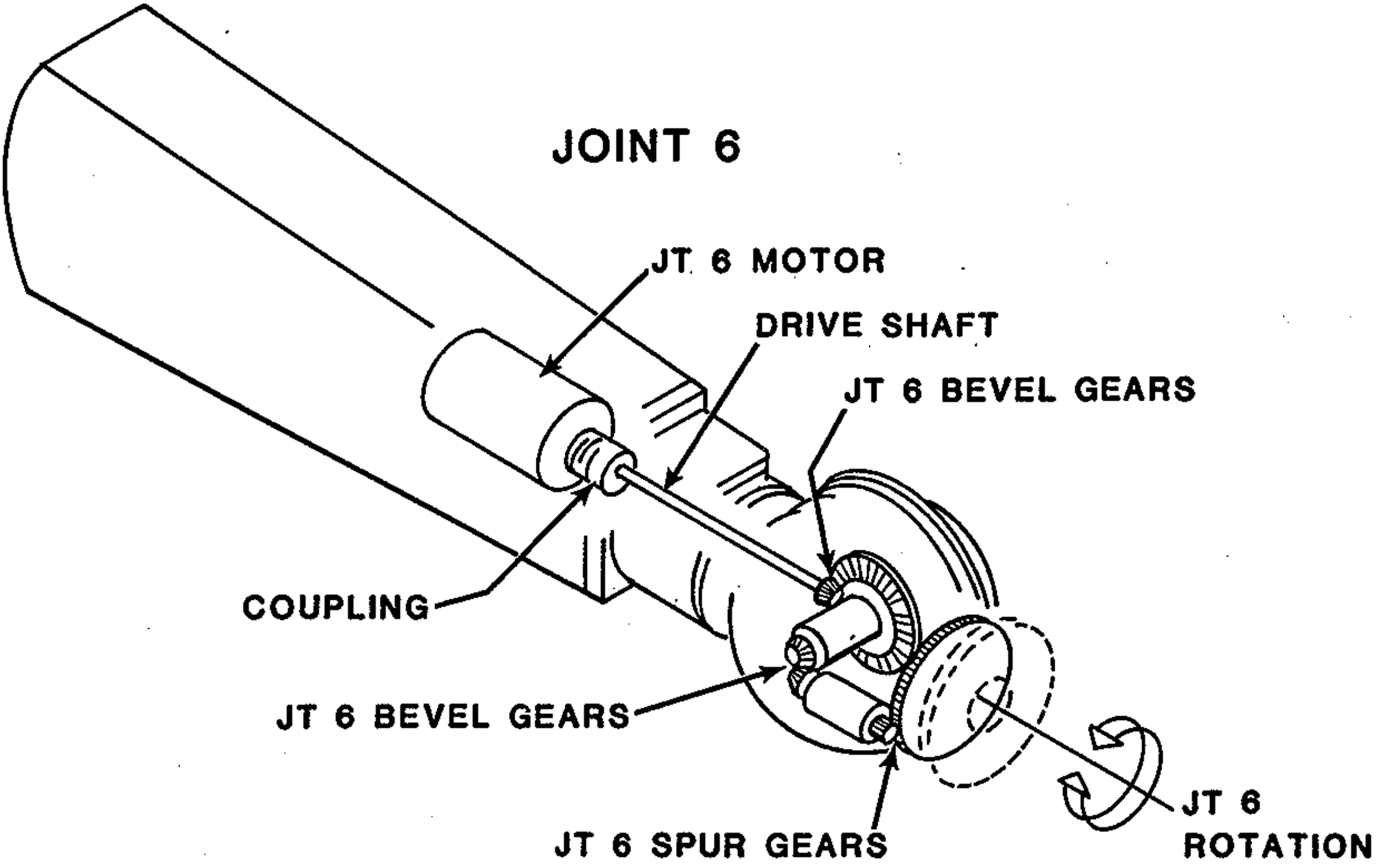


JOINT 3 MECHANICAL NOMENCLATURE IS IDENTICAL TO JOINT 2

JOINTS 4 & 5


JT 5 MOTOR
COUPLING
DRIVE SHAFT
JT 5 IDLER CARTRIDGE
JT 5 SPUR GEARS
JT 5 BEVEL GEARS
JT 5 ROTATION
JT 4 MOTOR
JT 4 SPUR GEARS
JT 4 IDLER CARTRIDGE
COUPLING
JT 4 IDLER CARTRIDGE
JT 4 SPUR GEARS
JT 4 ROTATION

JOINT 6


JT 6 MOTOR
DRIVE SHAFT
JT 6 BEVEL GEARS
COUPLING
JT 6 BEVEL GEARS
JT 6 SPUR GEARS
JT 6
ROTATION