## REFERENCES


1. Asada, H., Kanade, T., and Takeyama, I. "Control of a Direct-Drive Arm", *ASME Journal of Dynamic Systems, Measurement, and Control*, 105(3), 136-142, 1983.

2. Asada, H., and Slotine, J.J.E., *Robot Analysis and Control*, Wiley, New York, 1986.

3. Denavit, J., and Hartenberg, R.S., "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices", *ASME Journal of Applied Mechanics*, 22, 215-221, June 1955.

4. de Silva, C.W., "A Motion Control Scheme for Robotic Manipulators", *Proc. 1984 Canadian CAD/CAM and Robotics Conf.*, 13, 131-7, Toronto, June 1984.

5. _____, "Motion Sensors for Industrial Robots", *Mechanical Engineering*, ASME, 107(6), 40-51, June 1985.

6. _____, " Advanced Techniques for Robotic Manipulator Control", *Proc. 1986 Int. Congress on Tech. and Tech. Exchange*, 148-153, Pittsburgh, October 1986.

7. _____, *Control Sensors and Actuators* , Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1988.

8. de Silva, C.W. and Van Winssen, J.C., "A Recursive Algorithm for Least Squares Trajectory Control of Robotic Manipulators", *Proc. American Control Conf.*, 3, 1722-1727, Seattle, June 1986.

9. de Silva, C.W., Price, T.E., and Kanade, T., " Torque Sensor for Direct Drive Manipulators", *ASME Journal of Engineering for Industry*, 109(2), 122-127, May 1987.

10. de Silva, C.W. and Van Winssen, J.C., "Least Squares Adaptive Control for Trajectory Following Robots", *ASME Journal of Dynamic Systems, Measurement, and Control*, 109(2), 104-110, June 1987.

11. de Silva, C.W., Chung, C.L., and Lawrence, C., "Base Reaction Optimization of Robotic Manipulators for Space Applications", *Proc. Int. Symposium on Robots,* Sydney, November 1988 (In press).

12. Dubois, D. and Prade, H., *Fuzzy Sets and Systems*, Academic Press, Orlando, 1980.

13. Dubowski, S. and Des Forges, D.T., "The Application of Model-Referenced Adaptive Control to Robotic Manipulators", *ASME Journal of*

*Dynamic Systems, Measurement, and Control*, 101, 193-200, September 1979.

14. Francis, J.C. and Leitch, R.R., "ARTIFACT: A Real-time Shell for Intelligent Feedback Control", *Research and Development in Expert Systems*, Bramer, M.A. (ed.), Cambridge University Press, 1985.

15. Freedy, A., "Learning Control in Remote Manipulators and Robot Systems", *Learning Systems*, 53-69, American Control Council, New York, 1973.

16. Fu, K.S., Gonzalez, R.C., and Lee, C.S.G., *Robotics*, McGraw-Hill, New York, 1987.

17. Goff, K.W., "Artificial Intelligence in Process Control", *Mechanical Engineering*, ASME, 107(10), 53-57, October 1985.

18. Gupta, M.M. and Sanchez, E. (eds.), *Fuzzy Information and Decision Processes*, North-Holland, Amsterdam, 1982.

19. Hemami, H. and Camana, P.C., "Nonlinear Feedback in Simple Locomotion Systems", *IEEE Trans. on Automatic Control*, AC-21(6), 855-860, December 1976.

20. Hewit, J.R. and Burdess, J.S., "Fast Dynamic Decoupled Control for Robotics using Active Force Control", *Mechanism and Machine Theory,* 16(5), 535-542, 1981.

21. Hirota, K., Arari, Y., and Pedrycz, W., "Robot Control Based on Membership and Vagueness", *Approximate Reasoning in Expert Systems*, Gupta, M.M., et. al. (eds.), 621-635, Elsevier, Amsterdam, 1985.

22. Hollerbach, J.M., "A Recursive Formulation of Lagrangian Manipulator Dynamics", *Proc. Joint Automatic Control Conf.*, TP10-B, San Francisco, 1980.

23. Holmblad, I.P. and Ostergaard, J.J., "Fuzzy Logic Control: Operator Experience Applied in Automatic Process Control", *FLS Review*, Copenhagen, 1981.

24. Horn, B.K.P. and Raibert, M.H., "Manipulator Control Using the Configuration Space Method", *The Industrial Robot*, 5(2), 69-73, June 1978.

25. Isik, C. and Mystel, A., "Decision Making at a Level of Hierarchical Control for Unmanned Robots", *Proc. 1986 IEEE Int. Conf. on Robotics and Automation,* 1772-1778, IEEE Computer Society, Los Angeles, 1986.

26. Kahn, M.E., "The Near-Minimum-Time Control of Open-Loop Articulated Chains", *A.I. Memo 177*, Stanford AI Laboratory, California, December 1969.

27. Khosla, P., *Real-Time Control and Identification of Direct-Drive Manipulators,* Ph.D. Thesis, Dept. Elec. and Comp. Engineering, Carnegie Mellon University, Pittsburgh, August 1986.

28. Kornblugh, R.D., *An Experimental Evaluation of Robotic Manipulator Dynamic Performance under Model Referenced Adaptive Control*, S.M. Thesis, Dept. Mech. Engineering, Massachusetts Institute of Technology, Cambridge, September 1984.

29. Luh, J.Y.S., Walker, M.W., and Paul, R.P.C., "On-line Computation Scheme for Mechanical Manipulators", *ASME Journal of Dynamic Systems, Measurement, and Control*, 102, 69-76, June 1980.

30. Lynch, P.M., "Minimum-time Sequential Axis Operation of a Cylindrical Two-axis Manipulator", *Proc. Joint Automatic Control Conf.*, 1(WP-2A), Charlottesville, 1981.

31. Mason, M.T. and Salisbury, J.K., *Robot Hands and the Mechanics of Manipulation,* The MIT Press, Cambridge, 1985.

32. Mamdani, E.H., "Application of Fuzzy Logic to Approximate Reasoning using Linguistic Synthesis", *IEEE Trans. on Computers,* C-26(12), 1182-1191, December 1977.

33. Mamdani, E.H. and Gaines, B.R. (eds.), *Fuzzy Reasoning and Its Applications,* Academic Press, London, 1981.

34. *MUSE Support System Manual Set,* Cambridge Consultants Ltd., Cambridge, U.K., March 1987.

35. Paul, R., "The Mathematics of a Computer Controlled Manipulator", *Proc. Joint Automatic Control Conf.*, 1, 124-131, 1977.

36. _____, *Robot Manipulators,* The MIT Press, Cambridge, 1981.

37. Procyk, T.J. and Mamdani, E.H., "A Linguistic Self-Organizing Controller", *Automatica*, 15, 15-30, 1979.

38. *PROTUNER 1100 Instruction Manual Set,* Techmation Inc., Tempe, 1984.

39. Raibert, M.H. and Craig, J.J., "Hybrid Position/Force Control of Manipulators", *ASME Journal of Dynamic Systems, Measurement, and Control*, 102, 126-133, June 1981.

40. Scharf, E.M. and Mandic, N.J., "The Application of a Fuzzy Controller to the Control of a Multi-degree-of-freedom Robot Arm", *Industrial Applications of Fuzzy Control,* Sugeno, M. (ed.), North-Holland, Amsterdam, 41-61, 1985.

41. Shibly, H.A., *Performance Evaluation and Efficient Control of Trajectory Following Robots with Friction and Backlash,* Ph.D. Thesis, Dept. Mech. Engineering, Carnegie Mellon University, Pittsburgh, February 1988.

42. Slotine, J.J.E., "The Robust Control of Robot Manipulators", *The Int. Journal of Robotics Research,* 4(2), 49-64, Summer 1985.

43. Staugaard, A.C., *Robotics and AI,* Prentice Hall, Englewood Cliffs, 1987.

44. Sugeno, M. (ed.), *Industrial Applications of Fuzzy Control,* North-Holland, Amsterdam, 1985.

45. Tokumaru, H. and Iwai, Z., "Noninteracting Control of Nonlinear Multivariable Systems", *Int. Journal of Control,* 16(5), 945-958, 1972.

46. Tong, R.M., "A Control Engineering Review of Fuzzy Systems", *Automatica,* 13, 559-569, 1977.

47. Uicker, J.J., *On the Dynamic Analysis of Spatial Linkages using 4x4 Matrices,* Ph.D. Thesis, Northwestern University, Evanston, August 1965.

48. Van Amerongen, J., Van Nauta Lemke, H., and Van der Veen, J.C.T., "An Autopilot for Ships Designed with Fuzzy Sets", *Proc. Fifth IFAC/IFIP Int. Conf. on Digital Computer Appl. Process Control,* 1977.

49. Van Brussel, K. and Vastmans, L., "A Compensation Method for the Dynamic Control of Robots", *Proc. Conf. Robotics Research,* MS 84-487, Bethlehem, PA, August 1984.

50. Whitney, D.E., "Resolved Motion Rate Control of Manipulators and Human Prosthesis", *IEEE Trans. on Man-Machine Systems,* MMS-10(2), 47-53, June 1969.

51. _____, "Historical Perspective and State of the Art in Robot Force Control", *The Int. Journal of Robotics Research,* 6(1), 3-14, Spring 1987.

52. Wu, C.H. and Paul, R.P., "Resolved Motion Force Control of Robot Manipulators", *IEEE Trans. Systems, Man, and Cybernetics,* SMC-12(3), 266-275, May 1982.

53. Yoshikawa, T., "Analysis and Control of Robot Manipulators with Redundancy", *Robotics Research,* Brady, M. and Paul, R. (eds.), The MIT Press, Cambridge, 1984.

54. Young, K.K.D., "Controller Design for a Manipulator using Theory of Variable Structure Systems", *IEEE Trans. System, Man, and Cybernetics,* SMC-8(2), 101-109, February 1978.

55. Zadeh, L.A., "From Circuit Theory to System Theory", *Proc. Institute of Radio Engineers,* 50, 856-865, 1962.

56. Zimmerman, H.J., Zadeh, L.A., and Gaines, B.R. (eds.), *Studies in the Management Sciences*, North-Holland, Amsterdam, 1984.

# Appendix 1. Program Listing

Listings of the computer programs developed for the application described in Chapter 5 and Chapter 6, are given in this appendix. The programs included are the PopTalk code of the servo experts and fuzzy tuner, PopTalk interface programs, and the C program of the robot simulator.

```
/*************************************************************/
/*                                                           */
/*          THE KNOWLEDGE-BASED CONTROL PROGRAMS             */
/*                                                           */
/*************************************************************/


/*******************************************/
/*                                         */
/*      The definition of the schemas      */
/*                                         */
/*******************************************/

global 0 0 system 040886
~ object System_object_type ^System_object {
  system 100286 | FUBAR
~~declarations
    /* Forward declarations go in here */
~~internal_schemas
#1
~~schemas
~   collection generic_schema {
~     object generic_schema ^inference {
      desilva 120388 | Inferences of Joint Ruleset
~~      Comment
          Inferences made by the ruleset
          at each joint of robot regarding the error response.
~~      isa
          DB_obj
~~      slot name
          "noname"
~~      slot decay
          "pending"
~~      slot divergence
          "unknown"
~~      slot oscillation
          "unknown"
~~      slot offset
          "unknown"
~~      slot accuracy
          "unknown"
}
~     object generic_schema ^specs {
      desilva 120388 | Joint response specs.
~~      Comment
          Specifications on error decay,
          oscillation amplitude, and offset.
~~      isa
          DB_obj
~~      slot name
          "noname"
~~      slot lambda
          0.0
~~      slot ampl
          0.0
~~      slot ess
            0.0
}
~     object generic_schema ^error {
      desilva 120388 | Monitored error samples.
~~      Comment
          Sampling period and three
```

```
          successive values of maximum error, minimum error,
          and maximum
          error over three successive periods.
~~        isa
            DB_obj
~~        slot name
            "noname"
~~        slot period
              0.4
~~        slot e0
            0.0
~~        slot e1
            0.0
~~        slot e2
            0.0
}
}
~~instances
~   collection  {
}
~~libraries
~   collection Library_File {
~     object Library_File /usr/local/Muse/poptalklib
                          /sockets.pt {
      desilva 120688 | Pop source file
}
}
~~knowledge_sources
#2
~~notice_boards
~   collection NB {
~     object NB spec_nb1 {
      desilva 120388 | Joint 1 response specs.
~~      Comment
          Specifications on error response
          at joint 1 of robot.
~~      initial_entries
~         collection  {
~           object specs spec1 {
            desilva 120388 | Joint 1 response specs.
~~            Comment
                Specifications on error response
                at joint 1 of robot.
~~            lambda
                  1.0
~~            ampl
                  0.005
~~            ess
                  0.01
~~            name
                "spec1"
}
}
~~      demons
~         collection demon {
}
}
~     object NB spec_nb2 {
      desilva 120388 | Joint 1 response specs.
~~      Comment
            Specifications on error response
            at joint 2 of robot.
~~      initial_entries
~         collection  {
~     object specs spec2 {
            desilva 120388 | Joint 1 response specs.
```

```
~~              Comment
                    Specifications on error response
                    at joint 2 of robot.
~~              lambda
                    1.0
~~              ampl
                    0.005
~~              ess
                    0.01
~~              name
                    "spec2"
}
}
~~      demons
~         collection demon {
}
}
}
~~user_init
      lambda;
      /*    if $stream_from.schemaof /== stream then */
              socket_link();
      /*    endif; */
      endlambda,
}


/*****************************************/
/*                                       */
/*      The first knowledge source       */
/*                                       */
/*****************************************/

global 3 2 desilva 120388
~ object KS servo_ks1 {
  desilva 120388 | Joint 1 knowledge source.
~~  Comment
        The knowledge source that makes
        inferences on the dynamic response at
        joint 1 of robot.
~~  priority
      high_priority
~~  libraries
~     collection Library_File {
~       object Library_File chan1.pt {
        desilva 120688 | Pop source file
}
}
~~  initial_entries
~     collection  {
~       object inference infr1 {
        desilva 200688 | inference made on error
~~        Comment
              Inference made by servo expert at Joint 1.
~~        decay
            "pending"
~~        divergence
            "unknown"
~~        oscillation
            "unknown"
~~        offset
            "unknown"
~~        accuracy
            "unknown"
~~        name
            "infr1"
```

```
}
}
~~  demons
~     collection demon {
}
~~  relations
~     collection  {
}
~~  reasoning_modules
~     collection  {
#4
}
}


/************************************************/
/*                                              */
/*   The rule base of the first servo expert    */
/*   along with the fuzzy decision table        */
/*                                              */
/************************************************/

global 4 3 desilva 120388
~ object FPRuleset pid_rules1 {
  desilva 120388 | Joint 1 response testing rules.
~~  Comment
      Rules that make inferences on
      the joint 1 response of robot.
~~  rules
~     collection FPR {
~       object FPR okay_rule1 {
        desilva 120388 | Accurate response.
~~        Comment
            Checks whether the joint 1
            response is accurate.
~~        Source
              if
                  there is an inference I
                      -name "infr1" and
                  there is a specs
                      -name "spec1",
                -ess E and
                  there is an error ER
                      -name "error1",
                -e0 A where (abs(A) =< E),
                -e1 B where (abs(B) =< E),
                -e2 C where (abs(C) =< E)
              then
                  do(
                  vars strm1 = {stream:};
                  open('data1', 1) -> strm1;
                  strm1: flushit;
                  strm1: putchar(' 0.00  0.00  0.00');
                  strm1: flushit;
                  strm1: closeit; )     and
                  (printf('Joint 1 response is accurate.
                           \n');) and
                  assert {inference I: -accuracy "okay"}
                  and
                  delete ER from KS
}
~       object FPR oscillations_rule1 {
        desilva 120388 | Joint 1 oscillations.
~~        Comment
              Checks whether the error response
              at joint 1 has undesirable oscillations.
```

```
~~        Source
              if
                  there is an inference I
                      -name "infr1" and
                  there is a specs
                      -name "spec1",
                -ampl Am and
                  there is an error  ER
                      -name "error1",
                -e0 A,
                -e1 B,
                -e2 C where (((B < (A - 2.0 * Am)) and
                (C > (B + 2.0 * Am))) or
                ((B > (A + 2.0 * Am))
                and (C < (B - 2.0 * Am))))  then
                do(
                vars strm1 = {stream:};
                open('data1', 1) -> strm1;
                strm1: flushit;
                if abs(C - B) > 5.0 * Am
                then strm1: putchar('-0.50  0.00  0.50');
                else strm1: putchar('-0.30  0.00  0.30');
                endif;
                strm1: flushit;
                strm1: closeit; )  and
                  (printf('Joint 1 response has undesirable
                           oscillations. \n');) and
                  assert {inference I: -oscillation "present"}
                  and
                  delete ER from KS
}
~       object FPR nodecay_rule1 {
        desilva 120388 | Decay of Joint 1 error.
~~        Comment
            Checks whether the error response
            at joint 1 decays adequately.
~~        Source
              if
                  there is an inference I
                      -name "infr1" and
                  there is a specs
                      -name "spec1",
                -lambda L and
                  there is an error  ER
                      -name "error1",
                -period T,
                -e0 A,
                -e1 B,
                -e2 C where ((((A > 0.0) and (A > B)
                                and (B > C)) and
                ((B > (A * (1.0 - L * T))) or
                 (C > (A * (1.0 - 2.0 * L * T))))) or
                 (((A < 0.0) and (A < B) and (B < C)) and
                 ((B < (A * (1.0 - L * T))) or
                 (C < (A * (1.0 - 2.0 * L * T))))))  then

                 do(
                 vars strm1 = {stream:};
                 open('data1', 1) -> strm1;
                 strm1: flushit;
                 strm1: putchar(' 0.15  0.00  0.30');
                 strm1: flushit;
                 strm1: closeit; )  and
                  (printf('Joint 1 response does not decay
                           adequately. \n');) and
                  assert {inference I: -decay "unacceptable"}
```

```
                and
                delete ER from KS
}
~       object FPR divergence_rule1 {
        desilva 120388 | Divergence of Joint 1 error.
~~        Comment
            Checks whether the error
            response at joint 1 of robot diverges.
~~        Source
              if
                  there is an inference I
                      -name "infr1" and
                  there is an error  ER
                      -name "error1",
                -e0 A,
                -e1 B
                -e2 C where (((A >= 0.0) and (B > A)
                              and (C > B)) or
                      ((A =< 0.0) and (B < A)
                        and (C < B))) then

                      do(
                      vars strm1 = {stream:};
                      open('data1', 1) -> strm1;
                      strm1: flushit;
                      strm1: putchar('-0.30 -0.30  0.50');
                      strm1: closeit; )  and
                    (printf('Joint 1 error steadily diverges.
                            \n');) and
                    assert {inference I: -divergence "present"}
                    and
                    delete ER from KS
}
~       object FPR offset_rule1 {
        desilva 120388 | Offset in joint 1 response.
~~        Comment
              Checks whether there is a steady offset
              in the response at joint 1 of robot.
~~        Source
              if
                  there is an inference I
                      -name "infr1" and
                  there is a specs
                      -name "spec1",
                -ampl Am,
                -ess E and
                  there is an error  ER
                      -name "error1",
                -e0 A,
                -e1 B,
                -e2 C where (((A >= E) and (B >= E) and
                              (C >= E) and
                (A =< (E + Am)) and (B =< (E + Am)) and
                      (C =< (E + Am))) or
                 ((A =< (-1.0 * E)) and (B =< (-1.0 * E))
                 and
                 (C =< (-1.0 * E)) and (A >= -1.0 *(E + Am))
                 and
                 (B >= -1.0*(E + Am)) and (C >= -1.0*(E + Am))))
                 then

                 do(
                 vars strm1 = {stream:};
                 open('data1', 1) -> strm1;
                 strm1: flushit;
                 strm1: putchar(' 0.15  0.60  0.00');
```

```
                strm1: flushit;
                strm1: closeit; ) and
                 (printf('Joint 1 response has an unacceptable
                          offset. \n');) and
                 assert {inference I: -offset "present"}  and
                 delete ER from KS
}
~       object FPR final_rule1 {
        desilva 130688 | If other rules are not fired
~~        Comment
              Delete error object if rules are not fired
              on Joint 1.
~~        Source
              if ALONE
                    there is an error ER
                        -name "error1"  then

                    do(
                    vars strm1 = (stream:);
                    open('data1', 1) -> strm1;
                    strm1: flushit;
                    strm1: putchar(' 0.00  0.00  0.00');
                    strm1: flushit;
                    strm1: closeit; )     and
                    (printf('Rule search continued for Joint 1.
                             \n');) and
                    delete ER from KS
}
}
~~  interests
        [spec_nbl]
}


/*****************************************/
/*                                       */
/*     The second knowledge source       */
/*                                       */
/*****************************************/

global 5 2 desilva 230688
~ object KS servo_ks2 {
  desilva 120388 | Joint 1 knowledge source.
~~  Comment
        The knowledge source that makes
        inferences on the dynamic response at
        joint 2 of robot.
~~  priority
      high_priority
~~  libraries
~     collection Library_File {
~ object Library_File chan2.pt {
        desilva 120688 | Pop source file
}
}
~~  initial_entries
~     collection  {
~ object inference infr2 {
        desilva 200688 | inference made on error
~~        Comment
              Inference made by servo expert at Joint 2.
~~        decay
            "pending"
~~        divergence
            "unknown"
~~        oscillation
```

```
            "unknown"
~~          offset
              "unknown"
~~          accuracy
              "unknown"
~~          name
                "infr2"
}
}
~~  demons
~     collection demon {
}
~~  relations
~     collection  {
}
~~  reasoning_modules
~     collection  {
#6
}
}


/*****************************************************/
/*                                                   */
/*    The rule base of the second servo expert       */
/*    along with the fuzzy decision table            */
/*                                                   */
/*****************************************************/

global 6 5 desilva 230688
~ object FPRuleset pid_rules2 {
  desilva 120388 | Joint 1 response testing rules.
~~  Comment
        Rules that make inferences on
        the joint 2 response of robot.
~~  rules
~     collection FPR {
~ object FPR okay_rule2 {
        desilva 120388 | Accurate response.
~~        Comment
              Checks whether the joint 2
              response is accurate.
~~        Source
              if
                  there is an inference I
                      -name "infr2" and
                  there is a specs
                      -name "spec2",
                -ess E and
                  there is an error ER
                      -name "error2",
                -e0 A where (abs(A) =< E),
                -e1 B where (abs(B) =< E),
                -e2 C where (abs(C) =< E)  then

                  do(
                  vars strm2 = {stream:};
                  open('data2', 1) -> strm2;
                  strm2: flushit;
                  strm2: putchar(' 0.00  0.00  0.00');
                  strm2: flushit;
                  strm2: closeit; )     and
                  (printf('Joint 2 response is accurate.
                           \n');) and
                  assert {inference I: -accuracy "okay"}
                  and
```

```
                delete ER from KS
)
~ object FPR oscillations_rule2 {
        desilva 120388 | Joint 1 oscillations.
~~        Comment
              Checks whether the error response
              at joint 2 has undesirable oscillations.
~~        Source
              if
                  there is an inference I
                      -name "infr2" and
                  there is a specs
                      -name "spec2",
                -ampl Am and
                  there is an error  ER
                      -name "error2",
                -e0 A,
                -e1 B,
                -e2 C where (((B < (A - 2.0 * Am)) and
                (C > (B + 2.0 * Am))) or ((B > (A + 2.0 * Am))
                and (C < (B - 2.0 * Am))))  then

                do{
                vars strm2 = {stream:};
                open('data2', 1) -> strm2;
                strm2: flushit;
                if abs(C - B) > 5.0 * Am
                then strm2: putchar('-0.50  0.00  0.50');
                else strm2: putchar('-0.30  0.00  0.30');
                endif;
                strm2: flushit;
                strm2: closeit; )  and
                  (printf('Joint 2 response has undesirable
                          oscillations. \n');) and
                  assert {inference I: -oscillation "present"}
                  and
                  delete ER from KS
)
~ object FPR nodecay_rule2 {
        desilva 120388 | Decay of Joint 1 error.
~~        Comment
              Checks whether the error response
              at joint 2 decays adequately.
~~        Source
              if
                  there is an inference I
                      -name "infr2" and
                  there is a specs
                      -name "spec2",
                -lambda L and
                  there is an error  ER
                      -name "error2",
                -period T,
                -e0 A,
                -e1 B,
                -e2 C where ((((A > 0.0) and (A > B)
                              and (B > C)) and
                ((B > (A * (1.0 - L * T))) or
                 (C > (A * (1.0 - 2.0 * L * T))))) or
                 (((A < 0.0) and (A < B) and (B < C)) and
                 ((B < (A * (1.0 - L * T))) or
                 (C < (A * (1.0 - 2.0 * L * T)))))))    then

                 do{
                 vars strm2 = {stream:};
                 open('data2', 1) -> strm2;
```

```
                strm2: flushit;
                strm2: putchar(' 0.15  0.00  0.30');
                strm2: flushit;
                strm2: closeit; )    and
                 (printf('Joint 2 response does not decay
                          adequately. \n');) and
                 assert {inference I: -decay "unacceptable"}
                 and
                 delete ER from KS
}
~ object FPR divergence_rule2 {
        desilva 120388 | Divergence of Joint 1 error.
~~        Comment
              Checks whether the error
              response at joint 2 of robot diverges.
~~        Source
              if
                  there is an inference I
                      -name "infr2" and
                  there is an error  ER
                      -name "error2",
                -e0 A,
                -e1 B
                -e2 C where (((A >= 0.0) and (B > A) and
                              (C > B)) or
                      ((A =< 0.0) and (B < A) and (C < B)))
                      then

                      do(
                      vars strm2 = {stream:};
                      open('data2', 1) -> strm2;
                      strm2: flushit;
                      strm2: putchar('-0.30 -0.30  0.50');
                      strm2: flushit;
                      strm2: closeit; )   and
                  (printf('Joint 2 error steadily diverges.
                           \n');) and
                  assert {inference I: -divergence "present"}
                  and
                  delete ER from KS
}
~ object FPR offset_rule2 {
        desilva 120388 | Offset in joint 1 response.
~~        Comment
              Checks whether there is a steady offset
              in the response at joint 2 of robot.
~~        Source
              if
                  there is an inference I
                      -name "infr2" and
                  there is a specs
                      -name "spec2",
                -ampl Am,
                -ess E and
                  there is an error  ER
                      -name "error2",
                -e0 A,
                -e1 B,
                -e2 C where (((A >= E) and (B >= E) and
                              (C >= E) and
                (A =< (E + Am)) and (B =< (E + Am)) and
                (C =< (E + Am))) or
                 ((A =< (-1.0 * E)) and (B =< (-1.0 * E)) and
                 (C =< (-1.0 * E)) and (A >= -1.0 *(E + Am))
                 and
                 (B >= -1.0*(E + Am)) and
```

```
                (C >= -1.0*(E + Am)))  then

                do(
                vars strm2 = {stream:};
                open('data2', 1) -> strm2;
                strm2: flushit;
                strm2: putchar(' 0.15  0.60  0.00');
                strm2: flushit;
                strm2: closeit; )  and
                 (printf('Joint 2 response has an unacceptable
                         offset. \n');) and
                 assert {inference I: -offset "present"}  and
                 delete ER from KS
}
~ object FPR final_rule2 {
        desilva 130688 | If other rules are not fired
~~        Comment
              Delete error object if rules for Joint 2 are
              not fired.
~~        Source
              if ALONE
                    there is an error ER
                        -name "error2"    then

                    do(
                    vars strm2 = {stream:};
                    open('data2', 1) -> strm2;
                    strm2: flushit;
                    strm2: putchar(' 0.00  0.00  0.00');
                    strm2: flushit;
                    strm2: closeit; )     and
                    (printf('Rule search continued for Joint 2.
                             \n');) and
                    delete ER from KS
}
}
~~  interests
        [spec_nb2]
}
```

```
/************************************/
/*                                  */
/*     THE INTERFACE PROGRAMS       */
/*                                  */
/************************************/

/*****************************************************/
/*                                                   */
/*     The interface from the robot simulator        */
/*     to the first servo expert                     */
/*                                                   */
/*****************************************************/

vars error1 = {error:
                -name "error1",
                -period 0.4};
vars servo_ks1 ={KS:};

vars chan1 = constant {data_channel:
              -monitor_type DC_ALL,
              -monitor_demon lambda;
            error1:e1->error1:e0;
            error1:e2->error1:e1;
            chan1:value->error1:e2;
        servo_ks1:add(error1);
    printf(error1:e0, error1:e1, error1:e2, 'Joint 1
                                error = %, %, %, \n');
                          endlambda,
              -channel 1};


/*****************************************************/
/*                                                   */
/*     The interface from the robot simulator        */
/*     to the second servo expert                    */
/*                                                   */
/*****************************************************/

vars error2 = {error:
                -name "error2",
                -period 0.4};
vars servo_ks2 ={KS:};

vars chan2 = constant {data_channel:
              -monitor_type DC_ALL,
              -monitor_demon lambda;
            error2:e1->error2:e0;
            error2:e2->error2:e1;
            chan2:value->error2:e2;
        servo_ks2:add(error2);
    printf(error2:e0, error2:e1, error2:e2, 'Joint 2
                                error = %, %, %, \n');
                          endlambda,
              -channel 2};
```

```
/*********************************/
/*                               */
/*     THE ROBOT SIMULATOR       */
/*                               */
/*********************************/

#include <stdio.h>
#include <math.h>
#include "block_sockets.c"
#include "usocket.c"
float qd1, qd2, rdx, rdy, p=100.0;

main()
{
    float t, fi, rx, ry;
    float ei1=0.0, qdr1=0.0, q1;
    float qddr1, e1=0.05;
    float om1=1.0, ri1=0.0, taud1=0.01, delt=0.8;
    float dom1=0.0, dri1=0.0, dtaud1=0.0;
    float om1mx=2.0, ri1mx=0.1, taud1mx=1.0;
    float om1mn=0.5, ri1mn=0.0, taud1mn=0.0;
    float psen=5.0;
    float maxmin1,high1,low1;
    int n=500, edn=16, i=1, test=2;
    float distb1();
    float min(), max();

    float ei2=0.0, qdr2=0.0, q2;
    float qddr2, e2=0.05;
    float om2=1.5, ri2=0.0, taud2=0.01;
    float dom2=0.0, dri2=0.0, dtaud2=0.0;
    float om2mx=2.5, ri2mx=0.1, taud2mx=1.0;
    float om2mn=0.5, ri2mn=0.0, taud2mn=0.0;
    float maxmin2,high2,low2;
    float kinem(), distb2();

    FILE *fp, *fopen();

    fflush(stdout);
    muse_socket_to();

    rdx=1.0;
    rdy=0.25;
    kinem(0.0);
    q1 = qd1 - e1;
    q2 = qd2 - e2;

    maxmin1=e1;
    high1=e1;
    low1=e1;

    maxmin2 = e2;
    high2 = e2;
    low2 = e2;

    while (i <= n) {

    rx = 2.0 * cos(q1) + cos(q1 + q2);
    ry = 2.0 * sin(q1) + sin(q1 + q2);
    printf ("%8.4f\n", rx);
```

```
    printf ("%8.4f\n", ry);

    qddr1 = om1 * om1 * (e1 + ri1 * ei1 - taud1 * qdr1)
            + distb1(i);
    qdr1 = qdr1 + qddr1 * delt;
    q1 = q1 + qdr1 * delt;

    qddr2 = om2 * om2 * (e2 + ri2 * ei2 - taud2 * qdr2)
            + distb2(i);
    qdr2 = qdr2 + qddr2 * delt;
    q2 = q2 + qdr2 * delt;
(float) (fi = i);
    t = fi * delt;
    kinem(t);

    e1 = qd1 - q1;
    e2 = qd2 - q2;
    ei1 = ei1 + e1 * delt;
    ei2 = ei2 + e2 * delt;

/*    printf ("Joint error = "); */
/*    printf ("%8.4f\n", e1);    */
/*    printf ("%8.4f\n", e2);    */
/*    printf ("%8.4f\n", q1);    */
/*    printf ("%8.4f\n", q2);    */

    if (i % edn == 0) {
        if (maxmin1 == high1)
          maxmin1 = low1;
         else if (maxmin1 == low1)
          maxmin1 = high1;
         else if (test > 0)
          maxmin1 = high1;
         else
          maxmin1 = low1;

        sleep(2);
        muse_send_float(maxmin1, 1);
/*        printf ("%8.4f", maxmin1);    */

         fp = fopen("data1", "r");
       fscanf (fp, "%f%f%f", &dom1, &dri1, &dtaud1);
       fclose(fp);
/*       printf ("Joint 1 = %8.4f, %8.4f, %8.4f; ",
                  dom1, dri1, dtaud1); */

       om1 = om1 + dom1 * (om1mx - om1mn)/psen;
       ri1 = ri1 + dri1 * (ri1mx - ri1mn)/psen;
       taud1 = taud1 + dtaud1 * (taud1mx - taud1mn)/psen;

       if (om1 < om1mn)
         om1 = om1mn;
       if (om1 > om1mx)
         om1 = om1mx;
       if (ri1 < ri1mn)
         ri1 = ri1mn;
       if (ri1 > ri1mx)
         ri1 = ri1mx;
       if (taud1 < taud1mn)
         taud1 = taud1mn;
       if (taud1 > taud1mx)
         taud1 = taud1mx;
/*        printf ("Joint 1 PID = %8.4f, %8.4f, %8.4f\n",
                  om1, ri1, taud1); */

       if (maxmin2 == high2)
```

```
          maxmin2 = low2;
        else if (maxmin2 == low2)
          maxmin2 = high2;
        else if (test > 0)
          maxmin2 = high2;
        else
          maxmin2 = low2;

       sleep(2);
       muse_send_float(maxmin2, 2);
/*       printf ("%8.4f", maxmin2);   */

      fp = fopen("data2", "r");
      fscanf (fp, "%f%f%f", &dom2, &dri2, &dtaud2);
      fclose(fp);
/*      printf ("Joint 2 = %8.4f, %8.4f, %8.4f; ",
                dom2, dri2, dtaud2); */

      om2 = om2 + dom2 * (om2mx - om2mn)/psen;
      ri2 = ri2 + dri2 * (ri2mx - ri2mn)/psen;
      taud2 = taud2 + dtaud2 * (taud2mx - taud2mn)/psen;

      if (om2 < om2mn)
        om2 = om2mn;
      if (om2 > om2mx)
        om2 = om2mx;
      if (ri2 < ri2mn)
        ri2 = ri2mn;
      if (ri2 > ri2mx)
        ri2 = ri2mx;
      if (taud2 < taud2mn)
        taud2 = taud2mn;
      if (taud2 > taud2mx)
        taud2 = taud2mx;
/*       printf ("Joint 2 PID = %8.4f, %8.4f, %8.4f\n",
                om2, ri2, taud2);  */

       test = test * (-1);
       }

       high1 = max(maxmin1, e1);
       low1 = min(maxmin1, e1);

       high2 = max(maxmin2, e2);
       low2 = min(maxmin2, e2);

    ++i;
    }
        muse_close_socket();
}


float min(x, y)
float x, y;
{
    if (x < y)
        return (x);
    else
        return (y);
}


float max(x, y)
float x, y;
{
    if (x > y)
```

```
        return (x);
    else
        return (y);
}

float distb1(i)
int i;
{
if (i == 100)
   return (0.05);
if (i == 200)
   return (-0.05);
if (i == 300)
   return (0.1);
if (i == 425)
   return (-0.1);

    return (0.0);
}

float distb2(i)
int i;
{
if (i == 100)
   return (0.1);
if (i == 200)
   return (-0.1);
if (i == 300)
   return (0.05);
if (i == 425)
   return (-0.05);

    return (0.0);
}


/*****************************************/
/*                                       */
/*          Inverse Kinematics           */
/*                                       */
/*****************************************/

float kinem(t)
float t;
{
float trm, alph;

if (t < p) {
  rdx = 1.0;
  rdy = t * t /(2.0 * p * p) + 0.25;
};
if (t >= p && t < 2.0 * p)  {
  rdx = 1.0;
  rdy = (-1.0) * t * t /(2.0 * p * p) + 2.0 *
        t /p - 0.75;
};
if (t >= 2.0 * p && t < 3.0 * p) {
  rdy = 1.25;
  rdx = 0.5 * (t/p - 2.0) * (t/p - 2.0) + 1.0;
};
if (t >= 3.0 * p)   {
  rdy = 1.25;
  rdx = (-0.5) * (t/p - 2.0) * (t/p - 2.0) +
        2.0 * (t/p - 2.0);
};
```

```
trm = rdx * rdx + rdy * rdy;
qd2 = acos((trm - 5.0)/4.0);
alph = asin(sin(qd2)/sqrt(trm));
qd1 = atan(rdy/rdx) - alph;
return;
}

/*****************************************/
/*                                       */
/*  Inverse kinematics for the faster turn  */
/*                                       */
/*****************************************/

float kinem(t)
float t;
{
float trm, alph;

if (t < 1.5 * p) {
  rdx = 1.0;
  rdy = t * t /(3.0 * p * p) + 0.25;
};
if (t >= 1.5 * p && t < 2.0 * p)  {
  rdx = 1.0;
  rdy = (-1.0) * t * t /(p * p) + 4.0 * t /p - 2.75;
};
if (t >= 2.0 * p && t < 2.5 * p) {
  rdy = 1.25;
  rdx = (t/p - 2.0) * (t/p - 2.0) + 1.0;
};
if (t >= 2.5 * p)   {
  rdy = 1.25;
  rdx = (-1.0) * (t/p - 4.0) * (t/p - 4.0)/3.0 + 2.0;
};

trm = rdx * rdx + rdy * rdy;
qd2 = acos((trm - 5.0)/4.0);
alph = asin(sin(qd2)/sqrt(trm));
qd1 = atan(rdy/rdx) - alph;
return;
```