National Conference with International Participation

ENGINEERING MECHANICS 2008

Svratka, Czech Republic, May 12 – 15, 2008

THE POSITION CONTROL DESIGN OF A STEWART PLATFORM USING LQ DESIGN METHOD

L. Březina [*] , L. Houfek [**]

Summary: *Proposed article deals with a Stewart platform position control design which combines using of NI LabView for the control designing and Matlab for the model description. Controller is designed via LQ design method and Stewart platform model contains inverse kinematics of the machine. Particular phases of the designing are described and also some advantages of using such approach are mentioned.*

1. Introduction

Stewart platform (Fig. 1) is a 6 DOF parallel mechanism containing six extendible links and a movable platform. The single link extension is provided by a ball screw which is screwed out by a DC motor. This is how the position and the angle change of the platform are provided.

The presented control design is based on the idea of connecting DC motors state space models with the inverse kinematics of a Stewart platform model. The required platform position and angular deflection is transformed via the inverse kinematic model to the link extensions needed to reach this pose. This value is then the reference value for the controller.

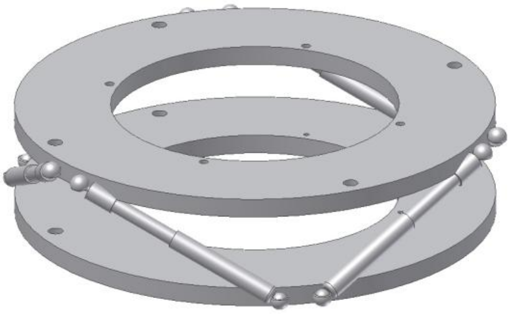


Fig.1 Stewart platform

[*] Ing. Lukáš Březina: Institute of Solid Mechanics, Mechatronics and Biomechanics, Faculty of Mechanical Engineering, Brno University of Technology; Technická 2, 616 69, Brno; Tel. +420 541 143 375; e-mail: ybrezi00@stud.fme.vutbr.cz

[**] Ing. Lubomír Houfek, Ph.D.: Institute of Solid Mechanics, Mechatronics and Biomechanics, Faculty of Mechanical Engineering, Brno University of Technology; Technická 2, 616 69, Brno; Tel. +420 541 142 887; e-mail: houfek@fme.vutbr.cz

2. Chosen approach

The whole scheme consists of three different parts. The first part is the Stewart platform inverse kinematic model (Merlet, 2006). The second part is the DC motor state space model and the last part contains the controller and observer design.

It is quite profitable to use NI LabView for the realization of the scheme. It has built in blocks like state feedback controller, linear quadratic regulator or predictive observer and it is easy to well organize the scheme. The possibility for later Software in the loop/ Hardware in the loop approach, where particular parts of the model are later substituted by real parts, is also great advantage.

It is also able to run Matlab mathematical software from within LabView. The inverse kinematic model was created right using this approach. Kinematics is made in Matlab, inputs (desired platform position and angular deflection) are taken from LabView and outputs (link extension) are linked back to LabView again. The advantage of this solution is the possibility of using Matlab commands for the kinematics design and at the same time using of the LabView blocks for designing of the control.

The DC motor model (Krause et al., 1995) was built in LabView, inputs are voltage and torque. The torque might be changed manually during the simulation to simulate changing load. The output of the model is the screwed out length of the ball screw. The DC motor state space model had originally angular velocity output. Then it was transformed to the screwed out length of the ball screw (link extension). The length is controlled according to the value of desired extension (output of inverse kinematics) by change of input voltage to the motor. The model is then transformed to the discrete model for the control purpose.

Then LQ design method (Franklin et al., 1980) was chosen for the state space feedback controller design. The optimal gain is computed according to R and Q matrices, where such parameters must be chosen to minimize the desired state. States estimations are provided by predictive observer.

2.1. Stewart platform inverse kinematics

Stewart platform inverse kinematics equations (1) give us information about how high have to be values of link extensions of the robot to reach desired position and orientation of the platform gravity center. Computed link extensions are then reference values for the controller.

$$\begin{aligned} & n_xP_{ix}+o_xP_{iy}+w_xP_{iz}+x-B_{ix}{}^{2}+\; n_yP_{ix}+o_yP_{iy}+w_yP_{iz}+y-B_{iy}{}^{2}+ \\ & +\; n_zP_{ix}+o_zP_{iy}+w_zP_{iz}+z-B_{iz}{}^{2}=S_i^2 \end{aligned} \tag{1}$$

for $i=1,...,6$.

Where P_{ix},P_{iy},P_{iz} are coordinates of link connection points on the platform, B_{ix},B_{iy},B_{iz} are coordinates of link connection points on the base, x,y,z are coordinates of the platform centre of gravity, S_i is the length of the i-th link and n,o,w are parts of rotation matrix (2)

$$R=\begin{bmatrix} \cos\psi\cos\nu & \text{-}\cos\psi\sin\varphi & \sin\psi \\ \sin\varphi\sin\psi\cos\nu+\cos\varphi\sin\nu & \text{-}\sin\varphi\sin\psi\sin\nu+\cos\varphi\cos\nu & \text{-}\sin\varphi\cos\psi \\ \text{-}\cos\varphi\sin\psi\cos\nu+\sin\varphi\sin\nu & \cos\varphi\sin\psi\sin\nu+\sin\varphi\cos\nu & \cos\varphi\cos\psi \end{bmatrix} \tag{2}$$

written as (3)

$$R = \begin{bmatrix} n_x & o_x & w_x \\ n_y & o_y & w_y \\ n_z & o_z & w_z \end{bmatrix}. \tag{3}$$

2.2. DC motor state-space model

Used state space model (4)

$$\begin{bmatrix} i' \\ \omega_2' \\ \varphi_2' \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & -\frac{K_b}{L}\frac{1}{K_G} & 0 \\ \frac{K_m}{J}K_G & -\frac{K_f}{J} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} i \\ \omega_2 \\ \varphi_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{K_G}{J} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ M_z \end{bmatrix} \tag{4}$$

$$x \;=\; 0 \quad 0 \quad K_S \begin{bmatrix} i \\ \omega_2 \\ \varphi_2 \end{bmatrix} + \; 0 \quad 0 \begin{bmatrix} u \\ M_z \end{bmatrix}$$

was derived from well known form (5).

$$\begin{bmatrix} i' \\ \omega_1' \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & -\frac{K_b}{L} \\ \frac{K_m}{J} & -\frac{K_f}{J} \end{bmatrix} \begin{bmatrix} i \\ \omega_1 \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} u \tag{5}$$

$$\omega_1 \;=\; 0 \quad 1 \begin{bmatrix} i \\ \omega_1 \end{bmatrix} + \; 0 \; u$$

Where $\omega_2 = K_G\omega_1$, $x = K_S\varphi_2$, R is the resistence of the circuit, L is the self-inductance of the armature, K_f is the linear aproximation of the viscous friction, K_b the emf constant, K_m the armature constant, K_G is the gear ratio between motor shaft and the ball screw, K_s is the screw pitch, i is electric current, ω_1 is the angular velocity on the motor shaft, ω_2 is the angular velocity on the ball screw, φ_1 is the orientation of the motor shaft, φ_2 is the orientation of the ball screw, u is the input voltage, M_z is the input torque and finally x is the link extension.

It was just necessary to transform original model to the model with the link extension as an output.

2.3. LQ design

The regulator design for a discrete system $u \;\; n \;\; = -Kx$ minimizes (6)

$$J = \sum \;\; x'Qx + u'Ru + 2x'Nu \tag{6}$$

subject to $x[n+1] = Ax[n] + Bu[n]$.

Our goal is to choose parameters $Q, R, \;\; N$ in such way to minimize J.

3. NI LabView implementation

3.1. Main block diagram

Main block diagram (fig. 2) consists of several parts. Whole Stewart platform kinematics is placed in "Matlab script node". This block provides running of Matlab on the LabView background, thus executing Matlab orders in LabView.

DC motor state-space model is then placed in more common "MathScript node". This is continuous model, so it is necessary to turn it to discrete. The model after the discretization is ready for the simulation.

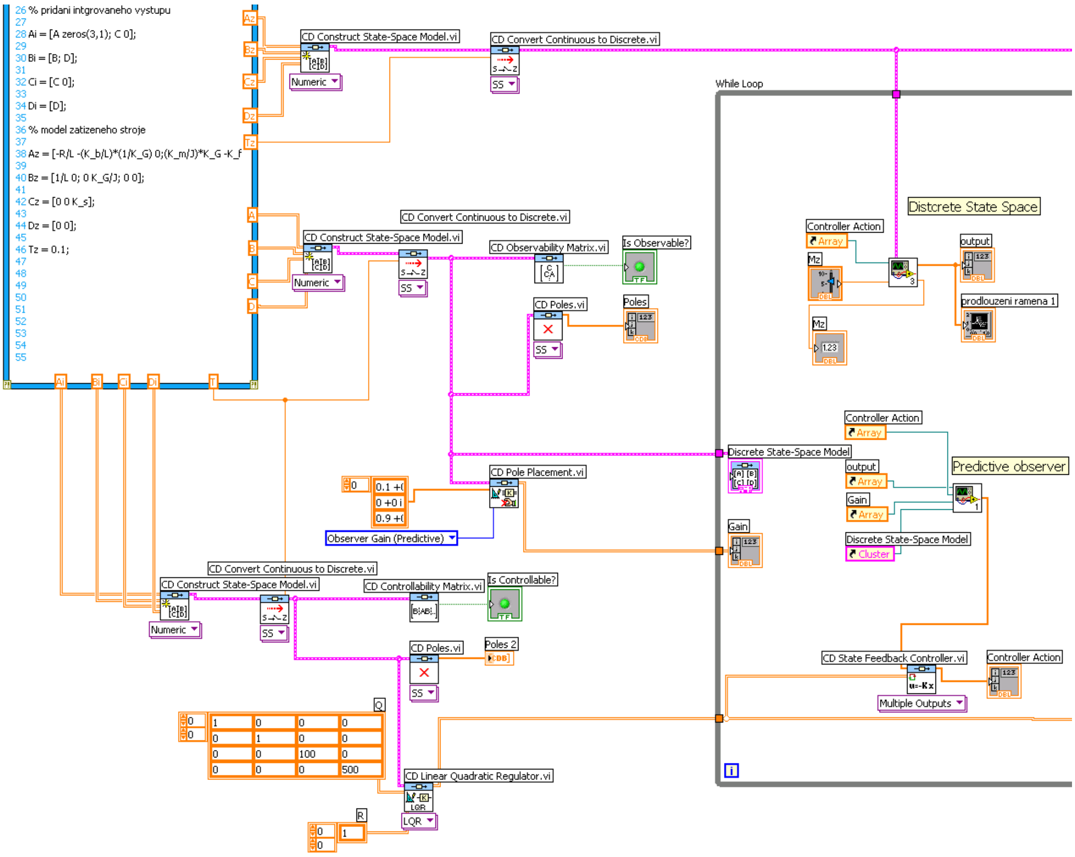
It is suitable to check the system (DC motor) poles for the observer designing and choose observer poles closer to zero. Such configuration guarantee that observer is fast enough to predict the system states. Again it is very easy to implement the observer because we have already built block for that (CD pole placement – predictive observer gain).

Designing of a linear quadratic regulator is also quite intuitive. Only thing we have to do is setting matrixes Q and R according to our requirements. In our case we need to minimize state φ_2. This is guaranteed by choosing high values in matrix Q on position $\;\; 3,3 \;\;$.

3.2. Front panel

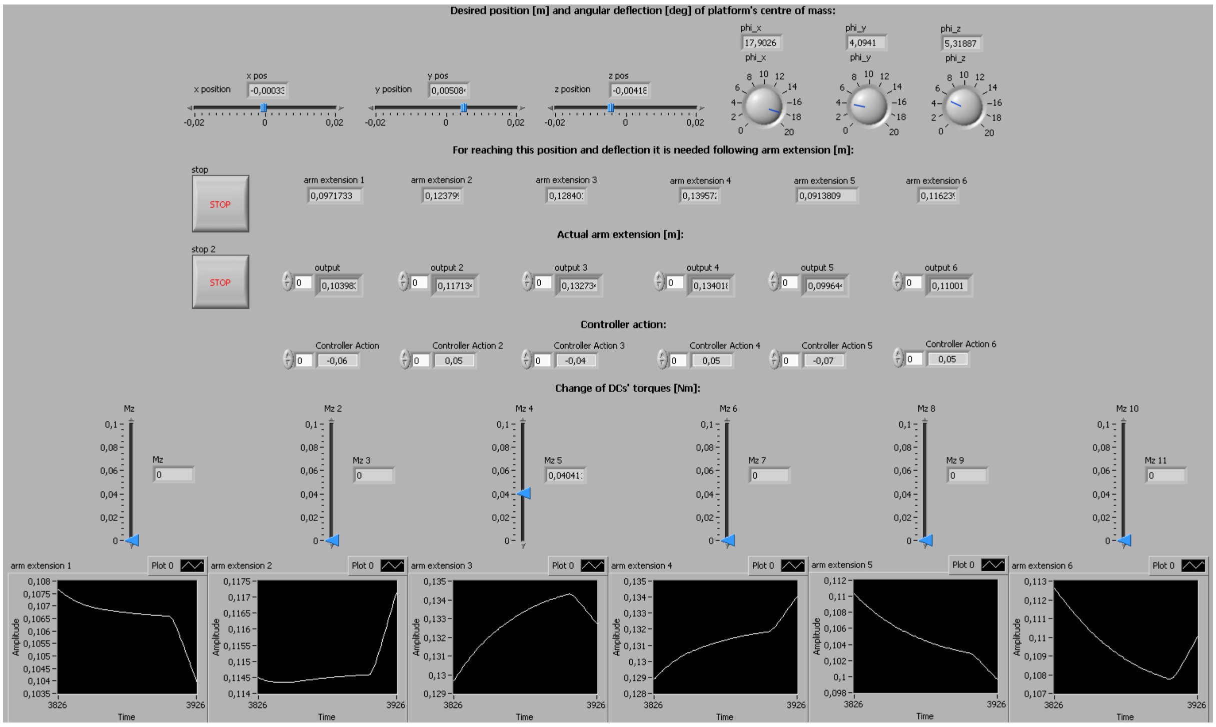
The most important thing that front panel (fig. 3) provides is the possibility of changing position and orientation of the platform. This can be done in real time while the simulation is running by six scroll bars (three for the position, three for the orientation).

New link extensions are computed through the inverse kinematics after changing the position and the orientation of the platform. These are desired values. We can also check actual values of the link extensions and controller actions needed to reach the desired length.



```
% pridani intgrovaneho vystupu

Ai = [A zeros(3,1); C 0];

Bi = [B; D];

Ci = [C 0];

Di = [D];

% model zatizeneho stroje

Az = [-R/L -(K_b/L)*(1/K_G) 0;(K_m/J)*K_G -K_f

Bz = [1/L 0; 0 K_G/J; 0 0];

Cz = [0 0 K_s];

Dz = [0 0];

Tz = 0.1;
```

Fig. 2 Main part of the block diagram



Desired position [m] and angular deflection [deg] of platform's centre of mass:

x pos: -0,00033 | y pos: 0,00508 | z pos: -0,00418

phi_x: 17,9026 | phi_y: 4,0941 | phi_z: 5,31887

For reaching this position and deflection it is needed following arm extension [m]:

arm extension 1: 0,0971733 | arm extension 2: 0,12379 | arm extension 3: 0,12840 | arm extension 4: 0,13957 | arm extension 5: 0,0913809 | arm extension 6: 0,11623

Actual arm extension [m]:

output: 0,10398 | output 2: 0,11713 | output 3: 0,13273 | output 4: 0,13401 | output 5: 0,09964 | output 6: 0,11001

Controller action:

Controller Action: -0,06 | Controller Action 2: 0,05 | Controller Action 3: -0,04 | Controller Action 4: 0,05 | Controller Action 5: -0,07 | Controller Action 6: 0,05

Change of DCs' torques [Nm]:

Mz: 0 | Mz 3: 0 | Mz 5: 0,04041 | Mz 7: 0 | Mz 9: 0 | Mz 11: 0

Fig. 3 Front panel

4. Simulation results

Outputs of the model are desired link lengths, so the main information we are interested in on the front panel is how close are actual link lengths to desired link lengths computed by inverse kinematics.

Desired values might be changed in real time. Following picture of actual link extension is for the desired value of 0,1524 m. The desired value was reached in c. 5 seconds.

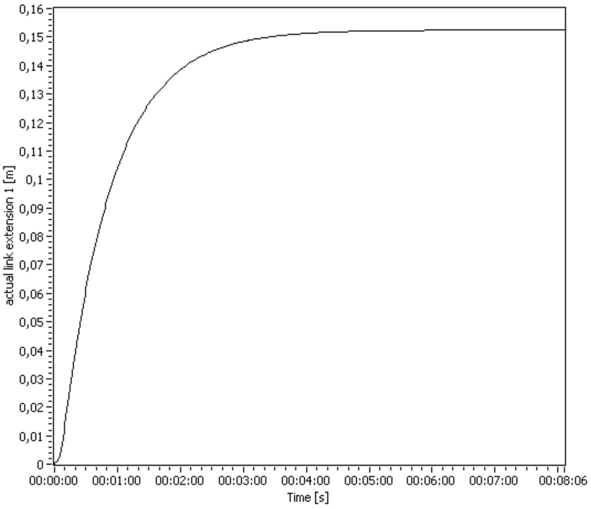


Fig. 4 Actual link extension for the desired value of 0,1524 m

So we can say that simulation results are satisfying. But we must keep on mind that whole simulation was done only for kinematic model of the Stewart platform. The model contained no masses and inertia moments.

5. Conclusions and future work

The Stewart platform position control design was presented in the paper. Also some general advantages of using NI LabView with Matlab cooperation when designing a control were mentioned.

In the future is expected modeling of the whole Stewart platform dynamics and simulating of control in Matlab and its toolboxes. Then tuned controller designed in Matlab will be transported to the LabView ready to be tested on the real machine.

Acknowledgement

This work is supported from research plan MSM 0021630518 Simulation modelling of mechatronic systems.

References

Franklin, G.F., J.D. Powell & M.L. Workman (1980) *Digital Control of Dynamic Systems (Second Edition)*. Addison-Wesley, Boston.

Krause, P. C. et al. (1995) *Analysis of Electric Machinery*. IEEE press, New York.

Merlet, J. P. (2006) *Parallel Robots (Second Edition)*. Springer, Dordrecht.