CONTROL OF LABORATORY MODEL BALL AND PLATE

Dr. Ing. Vratislav Hladký
Ing. Pavol Liščinský
Department of Cybernetics and Artificial Intelligence, FEI, TU Košice
Letná 9, 042 00 Košice, Slovak Republic
e-mail: vratislav.hladky@tuke.sk
e-mail: pavol.liscinsky@tuke.sk

Abstract

This article deals with the implementation of control algorithms for a laboratory model "ball and plate" with the aid of input/output card, type MF 614 and an application written in C/C++ programming language. The article also deals with the design and implementation of a computer vision algorithm for ball position recognition. A PD and limited integral term PID controllers are designed for the control of ball position. The proposed program solution utilizes multi-threading, one thread is used for ball position measurement, second for application of control action.

Key words: automatic control, trajectory tracking, ball and plate model, PID controller, computer vision in English language.

INTRODUCTION

Until now the programming instruments used for “ball and plate” model control included mostly MATLAB and SIMULINK. We assume that using control algorithms in C/C++ language might provide better insight into the problems encountered with this model and subsequently provide better control results compared to previous works [7, 10]. Flaws that caused problems with control, especially intermittent motion of the plate and also erroneous setting of plate tilt, which led to loss of information about the absolute plate tilt, were largely removed by the designed programs for plate tilt setting, using limitations of the speed of tilt change to a value that was tested multiple times.

MODEL DESCRIPTION

The "ball and plate" model is a two input (plate tilt in the X axis and in the Y axis), two output (ball position coordinates *X*, *Y*) system with a second order astatism. Model CE 151 Ball and Plate manufactured by Humusoft ltd. consists of three components: basic unit, camera and a lab card.

The basic unit contains several parts of importance, such as the plate, two step motors, which are used to tilt the plate and limit switches. The plate measures 400 × 400 mm. It is possible to manipulate the tilt of the plate to ±0,1878 rad from the horizontal position in both the *x* and *y* axes. To tilt the plate from the minimum value up to the maximum value approximately 600 steps are required, according to the manual [4]. The given value is inaccurate. The true number of steps required has been determined experimentally [8], 568 steps for the *X* axis and 526 steps for the *Y* axis. The value at horizontal position was determined only roughly, but sufficiently for the purposes of control.

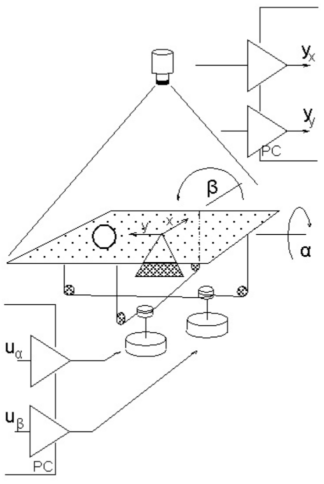

Fig. 1 Model diagram

In case that the plate reaches minimum or maximum tilt, the limit switches are triggered. The model contains no other sensors measuring absolute tilt of the plate. The limit switches are quite inaccurate. Repeated testing revealed that different number of steps were required to trigger the switches, which implies further measurement inaccuracies in the system.

According to the model manual the maximum frequency of control of the step motors is 8191 Hz. During testing it has been determined that as the control frequency of the steps surpasses cca 1200 Hz some of the steps are left out and absolute tilt information is lost. When using lower frequencies skipping of steps is quite rare. The designed application allows the user to specify the control frequency used.

CCD CAMERA AND MF614 CARD

The model includes a CCD camera, connected to the PC via a USB interface. The camera resolution is 160×220 pixels. The scanning frame rate is cca 30 frames per second. The image gained is processed with the aid of the OpenCV library. The cvQueryFrame function call, which enables camera frame capture into an IplImage type container, does not provide the frame instantly. The camera frame rate is not constant.If Figure or graph is wider than width of column it is recommended adapt size of figure that there will be not lot of free space on left and right and set wrapping text up and down. Figure description place to text field and merge with image (see Fig. 2).

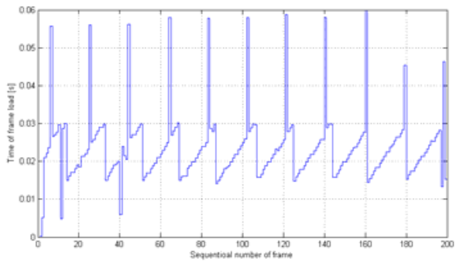

Fig. 2 Frame capture delay

The lab card, type MF 614, contains eight single-ended 12-bit analog input channels, four incremental input channels, four 12-bit output channels, eight digital inputs/outputs and four counters/timers [2]. The card is connected to the basic unit via X1 and X2 connectors. The X2 connector is used to connect counters/timers 0 to 3 with the model. Inputs and outputs of the counters/timers enable to count the number of PWM signals generated.

The X1 connector joins the END1 output, which indicates whether ES0 and ES1 limit switches are triggered and output END2, likewise indicating the triggering of ES2 and ES3 limit switches. This connection makes it clear that without knowledge of the direction of tilting the triggering of the limit switches does not provide us with the information whether the tilt is extreme in positive or negative sense. The connector includes two binary outputs, that set positive or negative direction of the motor's rotation.

The card is able to generate two PWM signals. The rising edge of the signal causes the execution of a single step of the motor in the set direction. It is possible to count the number of generated PWM pulses, which is necessary to determine the tilt of the plate.

PROGRAMMING SOLUTION OF PLATE POSITIONING AND BALL POSITION MEASUREMENT

The programming solution was created in the C/C++ programming language, using the MS Visual Studio 2005 environment.

The tilting of the plate is executed with the aid of two step motors. Speed of the steps depends on frequency of the PWM signal generated by the MF614 card. The card provides a maximum output frequency of 10 MHz [2]. The maximum stated control frequency of the motors is 8191 Hz [4]. When using such a frequency, information about some steps is lost, therefore high frequency is unsuitable. Significantly lower frequencies were used in the program design.

For accurate setting of the plate tilt to a desired position we have to know the present tilt. Since the basic unit contains only limit switches for extreme tilts in either direction, it is not possible to guarantee the starting plate position. When an extreme position is indicated by the limit switches, without prior knowledge of the direction of plate movement, or without an attempt to move the plate it is not possible to determine whether the plate reached an extreme tilt in positive or negative sense in a given axis. It is therefore necessary, before beginning of plate movement, to set the plate into an extreme position that we can guarantee. Such chosen extreme position will be considered a reference plate tilt and will be marked with [0,0] coordinates. The coordinates mark the number of steps of the motors taken from the reference tilt.

The algorithm of choice for tilting of the plate sets the tilt for both axes at the same time with a set duration. The first step, when changing the tilt, is the comparison of current plate tilt with the desired plate tilt. The result of this comparison is the setting of either positive or negative direction of rotation. Absolute difference of desired and current tilt is calculated and this difference determines the PWM signal frequency. Said frequency is given by division of difference of tilts by the set duration of execution of the desired plate tilt. After the direction and frequency has been set, the motion itself is carried out. Subsequently the number of steps executed in each axis is checked in a cycle. When the desired number of steps have been taken, or an extreme position has been reached, the tilting in the given axis is stopped by setting the PWM signal class to 0 and the motion is considered done. It is not possible to set a new desired tilt, until the previous desired or extreme tilt is reached.

The position of the ball on the plate is measured with the aid of a CCD camera connected to a PC. For image capture and recognition the OpenCV library, which contains efficient image capture and processing functions available in real time is utilized. This application used OpenCV version 2.1. In the complete control algorithm design an independent thread of the program was dedicated to image capture.

When designing the algorithm for ball position detection, a priori knowledge was used, which enabled usage of a simple and sufficient algorithm that is also relatively lightweight for computation.

The algorithm requires scanning the plate without the ball. This image of the plate represents background. During runtime the captured frame of the plate is converted to a grayscale representation and subsequently compared to a grayscale background image. Thresholding is applied on the comparison result. The threshold is used for removing noise, which comes from light reflecting from the plate's surface. Any remnant noise is removed by the morphological operation of closing that is erosion and subsequent dilation of the thresholded image. The processed image should only contain a silhouette of the ball inside the bounds of the plate without noise, which may however appear outside the bounds of the plate that are also captured, but unnecessary, therefore the algorithm works relative to a defined region of interest.

Region of interest is delimited by the upper left pixel of the displayed area, with coordinates [x_l, y_l] and lower right pixel, with coordinates [x_r, y_r]. It is assumed that the edges of the plate are parallel to the edges of the captured image. Boundaries of the region of interest were determined from the captured image. The values representing the upper right and lower left corner may be changed in the program as needed, for example due to camera movement.

Thanks to thresholdig we may view the resulting matrix as a matrix $\boldsymbol{M}$ (160x220 pixels) of logical values, where a value of 1 means the pixel corresponds to the ball. Across the entire specified region the number of pixels N belonging to the silhouette is counted as follows:

$$N = \sum_{i=x_l}^{x_r} \sum_{j=y_l}^{y_r} m_{ij} \tag{1}$$

where m_{ij} is an element of the matrix $\boldsymbol{M}$. The X axis coordinate of the ball is calculated by dividing the sum of the X axis coordinates of all the silhouette points by the number of silhouette points and the Y axis coordinate of the ball by dividing the sum of the Y axis coordinates of all the silhouette points by the number of silhouette points:

$$X = \sum_{i=x_l}^{x_r} \sum_{j=y_l}^{y_r} i.m_{ij} \tag{2}$$

$$Y = \sum_{i=x_l}^{x_r} \sum_{j=y_l}^{y_r} j.m_{ij} \tag{3}$$

In the program the acquisition of the X and Y coordinates is carried out by the following chunk of code from the CentroidROI() function [8].

The implementation of the code was designed and realized by methods and recommendations published in [1, 11]. Thus acquired position represents the coordinates of the pixel corresponding to the ball's center of mass. The value is scaled, so it falls into the interval <-1, 1>. The ball position coordinates are stored in a structure containing both scaled and raw values and also information on how up-to-date the position is. In case the ball position has not been recognized in a frame the setting of the plate tilt is not carried out.

THREAD USAGE

Due to the nature of actual plate tilt checking and the camera frame capture delay a sequential approach to the program would result in choppy plate movement and reduced quality of control. For these reasons an approach was proposed, where the main program splits into two separate threads. At program startup an instance of a GnaP class is created, which ensures initialization of the plate, the input files containing trajectory data and control parameters and output files, into which information about executed actions and data from the control process are added during runtime.

Next, threads TCamera and TAlgrth are created in suspended (paused) state. Afterwards TCamera thread is activated. After initialization procedures, TAlgrth thread is activated by user input. The program continues until the user requests to stop. In such an event the threads TCamera and TAlgrth are correctly stopped. Before program termination the plate is set to base position.

TCamera thread – the TCamera thread serves to detect ball position, compute the center of mass and display the captured and processed images. The availability of a frame is checked in a cycle. When a frame is not available, the camera state is set to "UNABLE" and at the next cycle condition check the cycle is terminated. If a frame is ready, its processing follows. In this processed image the center of mass of the ball is found and its coordinates calculated. The coordinates are stored via GnaP class methods into a private structure containing the coordinates of the pixel corresponding to the center of mass, the scaled coordinates usable in the algorithm and the information on the determined position recency. When storing a new position, the state of recency is set to "NEW_KOORD". This structure may be accessed only via a mutex, which ensures, that the structure is always being accessed by one thread only. The captured and processed images are shown in windows with the region of interest displayed. The processed (segmented) image also contains markers for computed current ball position and desired ball position.

TAlgrth thread – the second thread checks whether the data in the specified structure is up-to-date. In case the position is recent, the control action is computed by a given algorithm. The state of recency in the structure is set to "OLD_KOORD". The plate is tilted according to the computed control action. In case the position is not up-to-date, meaning that the movement according to the current data was finished before a new frame could be acquired, the thread is put to sleep for a short time, approximately half of the camera sampling period.

DEVISING CONTROL

When devising control we assume that it is possible to use two independent controllers, one for the X axis ball position control and second for the Y axis. We assume that mutual interactions for the proposed control scheme are insignificant.

The position algorithm was used for PID controller implementation. The first step is calculating the current error. The calculation of proportional, derivative and limited integral term follows. The control action computed in such a way is checked, so it would not cross the set limit, otherwise skipping of some steps might ensue, which would mean loss of absolute position information. After execution of the control action the next iteration of the algorithm follows.

The first implementation of the algorithm used only a PD controller, since the model is described as a system with second order astatism, however such a controller could not remove the steady-state error, therefore a PID controller was also devised. The first application did not include limits to the integral term and the wind-up effect was encountered during control. After applying limits to the integral term the steady-state error was removed as well as the wind-up effect.

The design of the controllers is based on a linear model defined in the model manual [3]. The model is described by a transfer function – for one axis of movement:

$$F_s = \frac{4,803}{s^2\left(0,1878s+1\right)} \qquad (4)$$

Naslin method was used for controller design [9]. The method was chosen based on the results of previous work with this model [6]. The devised controller was a PD controller with maximum overshoot of 5%. Computed values of the PD controller are: P = 0.7442, D = 0.5567 with the sampling period 0,1second.

As PD controller was not sufficient to eliminate the steady-state error of the system a PID controller was also devised, while integral term limiting was used to prevent wind-up effect. Computed values of the PID controller are: P = 0.7442, I = 0.4975, D = 0.5567 with the sampling period 0,1second.

Position algorithm was used for computing control action. The integral term was limited by setting a minimum and maximum value of the sum of errors. This led to better quality of control.

EXPERIMENTS – CONTROL TO REQUIRED VALUE

We did not design any simulational model for this work because devising of simulational models and experiments with them were parts of previous works. Acquired information from previous works was sufficient for design of controllers. We executed a number of experiments, where the main goals of these experiments was control of position of the ball and trajectory tracking. In the first group of experiments a position of the ball was controlled to the position [0,0]. In the second group of experiments we specified three types of trajectories to follow. For the control of position of the ball we used PID and PD controllers for both groups of experiments.

Required value of the ball position in the first group of experiments was [0,0]. The ball was manually disturbed two times and then performance of the controllers was observed.

During the first experiment PD controller was used. We applied Naslin's method to calculate parameters of the controller, and optional parameter α was set at 3% overshoot. In the second experiment we used PID controller and its parameters were again calculated by usage of the Naslin's method. Application of PID controller instead PD controller resulted in increasing of the quality of control and eliminated permanent control deviation (see Figure 4). Control was not stabilizing without limitation of influence of the integral part of the PID controller. Wind-up effect was observed in such case.

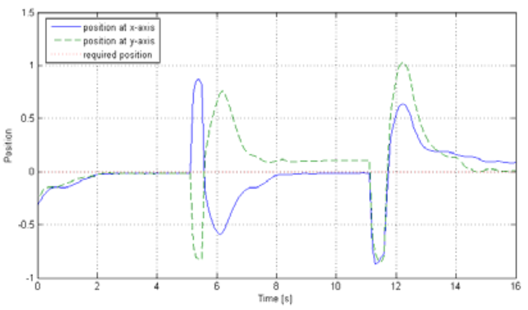

Fig. 3 Control to required value – PD Controller

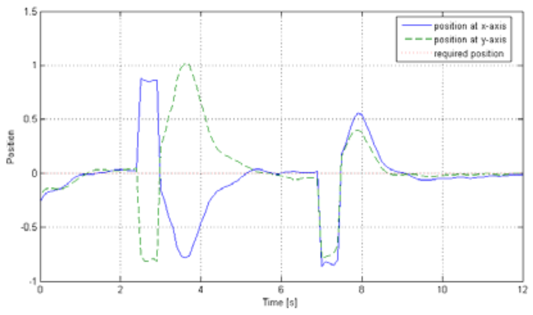

Fig. 4 Control to required value – PID Controller

EXPERIMENTS – TRAJECTORY TRACKING

For purposes of trajectory tracking we specified three types of the trajectories: square, 5-pointed star and circle (the experiments with a circle trajectory are described in [8]). The trajectories were generated using scripts written in Matlab and they were saved in a file containing a list of required positions and time values. Required positions were changing in preset frequencies.

Square trajectory – the square trajectory contains four points that represents corners of the square. The control process through the whole square takes approximately 20 seconds. Before beginning of the control the ball is set to the right down corner of the square. In this case if required position of the ball changes then it changes only one coordinate of the position. Because of this fact control should be performed mainly by one stepper motor. Interactions and non-linearities in the system should not be observed.

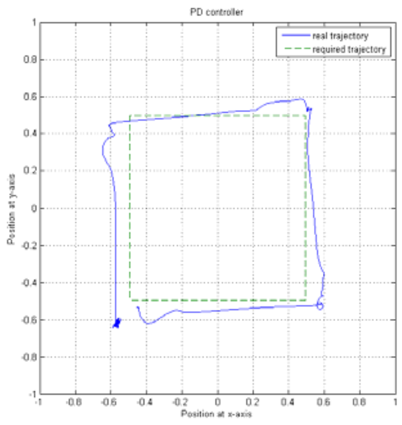

Fig. 5 Trajectory tracking – PD controller

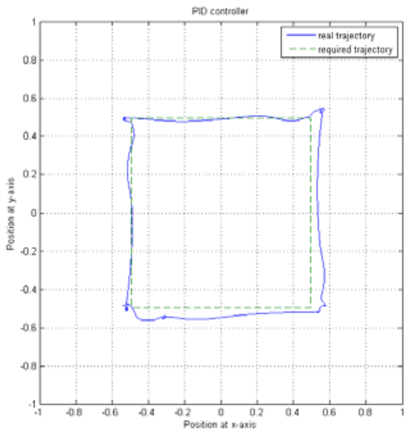

Fig. 6 Trajectory tracking – PID controller

Points of the trajectory were not chosen in the proximity of the plate edge because it is possible that overshoot during the control could cause ball beating off the plate edge and following process of control would take considerable time. Coefficients of the controller were again calculated by usage of the Naslin's method and optional parameter was set for 3% overshoot. In Figure 5 and Figure 6 we can see that PD controller was not able to eliminate the permanent control deviation. Higher quality of control was achieved by using PID controller where the control deviation was lower and control proces was faster.

5-pointed star trajectory – trajectory of 5-pointed star contains 5 peak points and position of the ball is set to the lower left point of the trajectory. Changing of the required position is very significant and both coordinates of the required position change. Quality of control was lower in this experiment than in previous two experiments. Control to the new required position has to be performed by both stepper motors simultaneously and non-linearity of the system in such case became evident intensively.

Control by application of PD controller did not cause significant overshoot unlike usage of PID controller. Subjectively higher quality of the control process was achieved by usage of PD controller in this case (see Figure 7 and Figure 8).

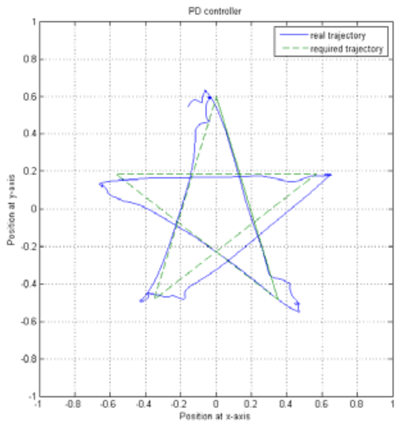

Fig. 7 Trajectory tracking – PD controller

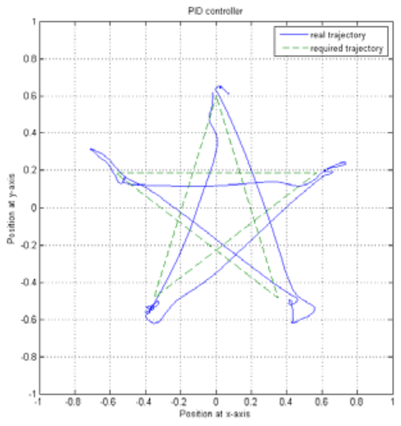

Fig. 8 Trajectory tracking – PID controller

RESULTS

For measuring of quality of the control process we decided to use some quantitative criterion of quality. This criterion was used to compare quality of control achieved by PD controller and PID controller. We applied the criterion represented by quadratic control area S_2 which is defined by the form [9]:

$$S_2 = \int_0^\infty \left(\Delta v(t)\right)^2 dt \qquad (5)$$

where $\Delta v(t)$ is the transitional part given as difference between real curve of the controlled parameter and steady-state value of the controlled parameter. The results were compared after calculation of both values of the criterion achieved by PD and PID controllers. Better results in control process at control to required value were achieved by using PID controller. PID controller was able to eliminate permanent control deviation.

In experiments with the trajectory tracking in the case of the square trajectory the value of the criterion achieved by PID controller was by 19% lower than the value achieved by PD controller. In the case of 5-pointed star trajectory the value of the criterion achieved by PID controller was by 12% lower than the value achieved by PD controller in spite of overshoot. However we can say that higher quality of the control was achieved by PID controller in all cases. The results of the experiments with the circle trajectory are described in [8].

CONCLUSION

The designed solution allows performing of the control of position of the ball without usage of Matlab and Simulink. The solution is easily modifiable. The solution allows application only simple types of controllers like PID controller and its variations. Results of experiments achieved by usage of designed solution were satisfying. Quality of control process achieved by usage of Matlab and Simulink was in some cases considerably lower. No identification of the model was done in this work. Identification of the model was part of multiple works before and information from this works was sufficient for design of controllers.

We believe this work explained some attributes of the model that caused problems at control by usage of Matlab and Simulink. Matlab allows simpler control of the model and implementation of different types of controllers than the control algorithm written in C/C++ programming language. Following works with the model could implement different types of controllers, for example state controller, fuzzy controller or others. Designed program application can be used in better identification of the model and its attributes, mostly those decreasing quality of control.

References

[1] Bradski, G., Kaehler, A.: Computer Vision with the OpenCV Library. O'Reilly Media, September, 2008. 580p. ISBN 978-0-596-51613-0.

[2] Humusoft: MF 614 Multifunction I/O, User´s Manual, 2002.

[3] Humusoft: CE151 Ball and Plate Apparatus, Educational Manual, 2004.

[4] Humusoft: CE151 Ball and Plate Apparatus, Technical Manual, 2004.

[5] Humusoft: Humusoft Data Acquisition Library Reference Manual, 2006.

[6] Jadlovská, A., Jajčišin, Š., Lonščák, R.: Modelling and PID Control Design of Nonlinear Educational Model Ball & Plate. In: Proceedings of the 17th International Conference on Process Control '09, June 9-12, 2009, Štrbské Pleso, Slovak Republic, Slovak University of Technology in Bratislava. pp.475-483. ISBN 978-80-227-3081-5.

[7] Kardoš, M.: Trajectory Tracking of Ball Movement in Laboratory Model Ball and Plate. Master’s Thesis, Technical University of Košice, 2008. 52p. (in Slovak)

[8] Liščinský, P.: Realization of control algorithms for laboratory model „ball and plate“ in C language. Master Thesis, Technical University of Košice, 2012. 53p. (in Slovak)

[9] Madarász, L., Főző, L., Andoga, R., Bučko, M.: Fundamentals of Automatic Control, the 2nd edition, Elfa, Košice, 2010. 401p. ISBN 978-80-8086-162-9. (in Slovak)

[10] Popovič, Ľ., Hladký, V., Sarnovský, J.: Modelling and Control of Educational Model Ball and Plate by Utilization of MPT Toolbox. In: Proceedings of the International Conference on Cybernetics and Informatics, February 10–13, 2010, Vyšná Boca, Slovak Republic, Slovak University of Technology in Bratislava, CD–ROM. 9p. ISBN 978-80-227-3241-3. (in Slovak)

[11] Zolotová, I., Karch, P.: Contribution to Modification of Graph Cut Method and Its Implementation in the Image Segmentation. International Journal of Circuits, Systems and Signal Processing. Vol. 6, no. 1, 2012. pp.49-56. ISSN 1998-4464.

This work was supported by Scientific Grant Agency of Ministry of Education, Science, Research and Sport of Slovak Republic and Academy of Science of Slovak Republic under grants VEGA no. 1/0298/12 (50%) and KEGA no. 021TUKE-4/2012 (50%).