Designing Internal-External Control Method for DeltaRobot Prototype to Manipulate Non-Linear Movement Object

by Dr. Aris Triwiyatno, S.t., M.t.

Submission date: 07-Feb-2020 09:25AM (UTC+0700)
Submission ID: 1252911349
File name: Paper_C-2-5.pdf (536.01K)
Word count: 2687
Character count: 14157

Designing Internal-External Control Method for Delta Robot Prototype to Manipulate Non-Linear Movement Object

Aris Triwiyatno, Muhammad Fikko Fadjrimiratno, Sumardi

muhammadfikko.fadjrimiratno.id@ieee.org

I. Introduction

Robot manipulator has increased quality and speed of manufactur proccess in most country. One of its configuration is Delta parallel robot, the most successfull and most commonly used as a pick and place robot [1]. This robot is most notable for its high accuracy and high speed.

For the Delta robot, the true end-effector position similarity with inverse kinematics generated position is uncertain, ultimately if its structural accuration is not good enough, e.g. joints and arms structure. Therefor, several researches have been conducted to anticipate this problem, such as the research to design a guaranteed accurate control scheme or to design calibration method. For examples, Olsson [2] calculated the acceleration and velocity of each joint to ensure trajectory accuration. Staicu and Ciocardia [3] included the dynamics of Delta robot in real time to generate needed moment and torque value to move end-effector to desired position. Lopez et al. [4] analyzed the singularity problem to avoid undesired posture of the arms. Noshadi et al. [5] implemented Fuzzy Learning Resolved Acceleration Control and Active Force Control to generate correct movement and torque for the Delta robot to ensure end-effector position accuration and its robustness from dynamics disturbances. Szep et al. [6] analyzed the kinematics and the workspace of the Delta robot to improve its design, trajectory planning, and control. Maurin [7] offered the calibration method to compensate accuracy degradation that was caused by joint friction from its continuously working loads.

However, none of these methods that have implemented real-time visual object tracking, thus not yet possible to manipulate object with non-linear movement. Fakhry and Wilson [8] have conducted similar research to detect object in real time, but it needed a wide viewing angle and that method was not implemented for Delta robot. Regarding to the above matters, the authors designed and implemented the method for the Delta robot to manipulate non-linear movement object based on the combination of internal control and external control.

II. Methods

A. Inverse Kinematics

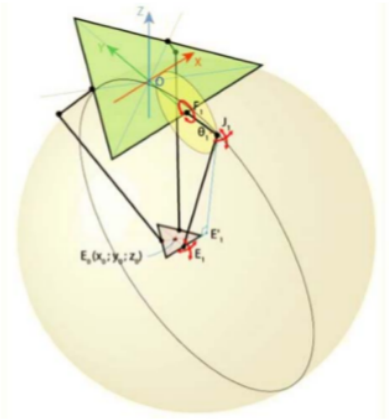


Fig. 1. Joints Movement of the Delta Robot [9]

Delta robot is a parallel based manipulator. Therefore, the kinematics approach is different from serial manipulator [10]. Here, we used intersection of circle and sphere that were formed

978-1-5090-0890-2/16/$31.00 ©2016 IEEE

by the movement of joints (Fig. 1). From Fig. 1, the F_iJ_i only rotate in *yz* axis, forming a circle that centered on F_i with the radius of r_f. Opposite from F_i joint, J_iE_i joint can freely rotate towards E_i because the use of universal joint that forming a sphere that centered on E_i with the radius of r_e [11]. The intersection between the sphere and *yz* axis forming a circle that centered on E'_i with the radius of E'_iJ_i, where E'_i is an E_i projection on *yz* axis. J_i can be determined as an intersection between two circles that their radius are centered on E'_i and F_i. If J_i has been determined, we can calculate θ_1.

From above principle, we can calculate inverse kinematics. Given the coordinates of end-effector as follows:

$$E(x_0, y_0, z_0) \tag{1}$$

$$E_1(x_0, y_0 - \frac{e}{2\sqrt{3}}, z_0) \tag{2}$$

$$E'_1(0, y_0 - \frac{e}{2\sqrt{3}}, z_0) \tag{3}$$

Then, determined coordinate of F_i :

$$F_1(0, -\frac{f}{2\sqrt{3}}, 0) \tag{4}$$

From the geometry of base and platform, defined several additional parameters as follows :

$${}^{P}\mathrm{P}_1 = \begin{Bmatrix} 0 \\ -U_P \\ 0 \end{Bmatrix};\ {}^{P}\mathrm{P}_2 = \begin{Bmatrix} \frac{S_P}{2} \\ W_P \\ 0 \end{Bmatrix};\ {}^{P}\mathrm{P}_3 = \begin{Bmatrix} -\frac{S_P}{2} \\ W_P \\ 0 \end{Bmatrix} \tag{5}$$

$$W_B = \frac{\sqrt{3}}{6}S_B\ ;\ U_B = \frac{\sqrt{3}}{3}S_B \tag{6}$$

$$W_P = \frac{\sqrt{3}}{6}S_P\ ;\ U_P = \frac{\sqrt{3}}{3}S_P \tag{7}$$

Defined :

$$\begin{cases} {r_f}^2 = F_1J_1 \\ \sqrt{{r_e}^2 - {x_0}^2} = E'_1J_1 \end{cases} \tag{8}$$

The coordinates from (1) – (4) are inputted :

$$\begin{cases} {r_f}^2 = (xF_1, yF_1, zF_1)(xJ_1, yJ_1, zJ_1) \\ \sqrt{{r_e}^2 - {x_0}^2} = (xE'_1, yE'_1, zE'_1)(xJ_1, yJ_1, zJ_1) \end{cases} \tag{9}$$

$$\begin{cases} {r_f}^2 = (xJ_1 - xF_1)^2 + (yJ_1 - yF_1)^2 + (zJ_1 - zF_1)^2 \\ \sqrt{{r_e}^2 - {x_0}^2} = (xE'_1 - xJ_1)^2 + (yE'_1 - yJ_1)^2 + (zE'_1 - zJ_1)^2 \end{cases} \tag{10}$$

$$\begin{cases} {r_f}^2 = \left(yJ_1 + \frac{f}{2\sqrt{3}}\right)^2 + (zJ_1)^2 \\ \sqrt{{r_e}^2 - {x_0}^2} = \left(yJ_1 - y_0 + \frac{e}{2\sqrt{3}}\right)^2 + (zJ_1 - z_0)^2 \end{cases} \tag{11}$$

$$\begin{cases} (zJ_1)^2 = {r_f}^2 - \left(yJ_1 + \frac{f}{2\sqrt{3}}\right)^2 \\ \left(yJ_1 - y_0 + \frac{e}{2\sqrt{3}}\right)^2 = \sqrt{{r_e}^2 - {x_0}^2} - (zJ_1 - z_0)^2 \end{cases} \tag{12}$$

Then, θ_1 can be found by with following equation :

$$\theta_1 = tan^{-1}\left(\frac{z_{J_1}}{y_{F_1} - y_{J_1}}\right) \tag{13}$$

To find θ_2 dan θ_3, rotation matrix is done by 120° on E'_i, E_i , dan F_i , *i*=2,3 as follows [2] :

$${}^{R}_{i}\mathrm{R} = \begin{Bmatrix} cos \propto_i & -sin \propto_i & 0 \\ sin \propto_i & cos \propto_i & 0 \\ 0 & 0 & 1 \end{Bmatrix}\ i=2,3 \tag{14}$$

From (14), next arm coordinates can be converted as follows

$$x_2 = (x_0 \cos(120)) + (y_0 \sin(120)) \tag{15}$$

$$y_2 = -(x_0 \sin(120)) + (y_0 \cos(120)) \tag{16}$$

$$x_3 = (x_0 \cos(240)) - (y_0 \sin(240)) \tag{17}$$

$$y_3 = (x_0 \sin(240)) + (y_0 \cos(240)) \tag{18}$$

Coordinates of *z* axis are not converted because the *z* coordinates will be the same on each arm. Next, the converted coordinates are inputted to (12) and arm degree position can be found with (13).

B. Design of the Delta Robot Prototype

Prototype of Delta robot was needed to implement the designed method (Fig. 2). Geometric parameters of the robot are defined with the parameters that will be needed to move the Delta robot through inverse kinematics. Those parameters are as follows [12][13]:

1. W_B = The length between center of gravity of the base into the actuator axis
2. U_P = The length between center of gravity of the platform into the center point axis of passive joint that link the parallelograms with the platform
3. L = The length of the parallelograms
4. l = The length of the actuator arm

From (6) and (7), to find W_B and U_P, S_B (the length of triangle side of base sector) and S_P (the length of triangle side of platform sector) are needed. S_B and S_P values are defined as follows :

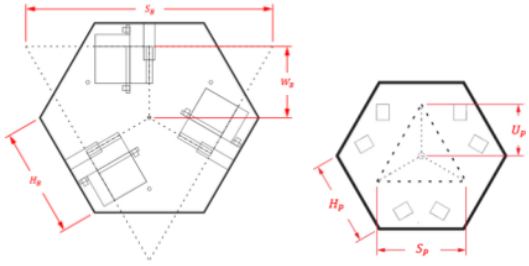
$$S_B = 200\ mm$$
$$S_P = 42.6\ mm$$

Then,

$$W_B = \frac{\sqrt{3}}{6}S_B = \frac{\sqrt{3}}{6}200 = 57.7\ mm$$
$$U_P = \frac{\sqrt{3}}{3}S_P = \frac{\sqrt{3}}{3}42.6 = 24.6\ mm$$

L and l are defined as follows :

$$L = 52\ mm$$
$$l = 175\ mm$$



(a) (b)

Fig. 2 Geometric design of (a) *Base* and (b) *Platform*

With defined parameters, the 3D design of the Delta robot prototype was made as shown in Fig. 3 and Fig. 4.

Fig. 3 Trimetic view of Delta robot prototype 3D design

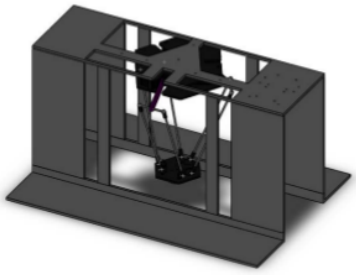


Fig. 4 3D design of the Delta robot prototype hanger

From designed Delta robot prototype, block diagram of the hardware is shown in Fig. 5. The inputs are potensiometers and Logitech c270h webcam. Controller hardwares are PC (Personal Computer) and ATMega16A microcontroller. Those two controllers communicate with K125R serial interface. The outputs are three low powered DC motor.

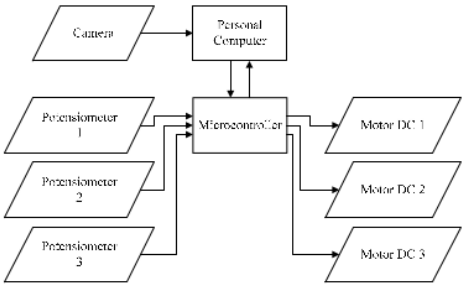


Fig. 5 Block diagram of the Delta robot prototype

Software is implemented on PC and microcontroller. PC software was programmed with Microsoft Visual Studio 2008 and AForge.NET library. PC software was programmed to process the digital image with RGB filter and blob counting, external control, inverse kinematics calculation, and Human Machine Interface (HMI). On microcontroller, the software was programmed with CodeVision AVR 2.05.3. Microcontroller software was programmed to read arm angle with potentiometer, to process internal control, and to generate an actuation signal in the form of PWM (Pulse Width Modulation) to control DC Motor.

C. Control Scheme

The used internal control is a combination of RMAC (Resolved Motion Acceleration Control) and PID (Proportional Integral Derivative) or Hybrid RMAC-PID. RMAC was invented by Luh et al. [14] as an alternative method to control dynamics system, such as robot manipulator, the system that well known for its nonlinearities. RMAC was developed from RMRC (Resolved Motion Rate Control) with including acceleration component so that the robot movement would be smoother. The output equation of the RMAC controller is as follows :

$$u(t) = kp\left(\theta_{ref}(t) - \theta_{act}(t)\right) + kv\left(\dot{\theta}_{ref}(t) - \dot{\theta}_{act}(t)\right) + ka\left(\ddot{\theta}_{ref}(t) - \ddot{\theta}_{act}(t)\right) \quad (19)$$

From (19), the references are θ_{ref} (position reference), $\dot{\theta}_{ref}$ (velocity reference), and $\ddot{\theta}_{ref}$ (acceleration reference). θ_{ref} was obtained from inverse kinematics. $\dot{\theta}_{ref}$ was from the θ_{ref} derivation. As a purpose of the acceleration control, then when the robot is moving, acceleration is expected to be zero, means that disturbance from outside that interference with acceleration can be kept to be zero [15]. Thus, $\ddot{\theta}_{ref}$ is referenced by zero. Next, θ_{act} (real position) can be directly measured by potentiometer.

The PID control system just needs the reference of θ_{ref} and eliminator components of *kp, ki,* and *kd*. Because *kp* has included on RMAC, then on hybrid RMAC-PID scheme, only *ki* and *kd* that were added. The output of this control scheme is *u*. *u* is the PWM value that control DC motor. Actuation signal *u* is defined as follows :

$$u(t) = kp\left(\theta_{ref}(t) - \theta_{act}(t)\right) + kv\left(\dot{\theta}_{ref}(t) - \dot{\theta}_{act}(t)\right) + ka\left(\ddot{\theta}_{ref}(t) - \ddot{\theta}_{act}(t)\right) + ki\left(\int\left(\theta_{ref}(t) - \theta_{act}(t)\right)dt\right) + kd\left(\frac{d\left(\theta_{ref}(t) - \theta_{act}(t)\right)}{dt}\right) \quad (20)$$

On the external control, PID control was used. The differences from PID on internal control are on the inputs and outputs. The needed inputs are x_{ref} (x axis coordinate of the object, x_{act} (x axis coordinate of the end-effector), y_{ref} (y axis coordinate of the object), and y_{act} (y axis coordinate of the end-effector). The output is the moving task in the form of the x axis and the y axis increasing or decreasing. That increasing or decreasing position was converted into angle values with inverse kinematics and became the references for internal control. This relationship between internal control and external control was forming a unified internal-external control (Fig. 6).

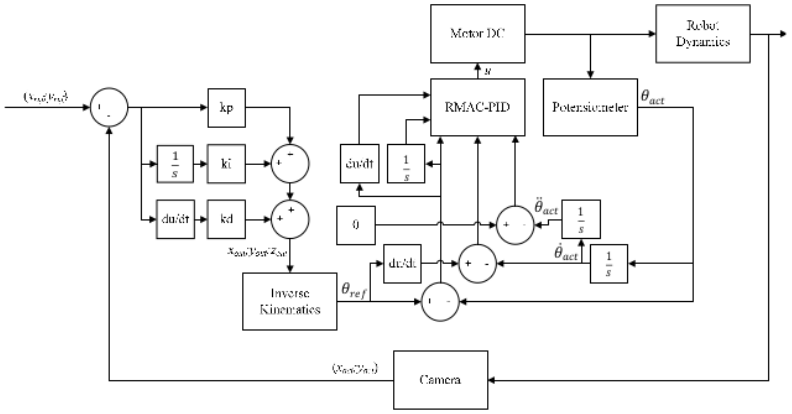


Fig. 6 The unified internal-external control scheme

III. RESULTS

A. Potentiometer Calibration Result

Based from calibration that was done on each potentiometer on each robot arm, Table. 1, Table. 2, and Table. 3 show that each potentiometer has its own different characteristics. It is seen from ADC (Analog to Digital Converter) and voltage value that are different for each potentiometer for the same angle position of the robot arm. That was because of the low quality potentiometer and can be replaced with a high quality rotary encoder that priced much higher. But, these different characteristics can be solved by implementing separated control schemes for each robot arm mechanism. However, this low quality potentiometers are still interfering with accuration because of its low linearity, resulting in the ADC value that is frequently changing. To solve it, Kalman filter will be used in the next research.

Tabel 1 Calibration result of the first potentiometer

Arm 1			
	Angle (Degree)	**VPout (Volt)**	**ADC Value (Decimal)**
Minimum Position	-60	2,35	481
Center Position	0	1,42	290
Maximum Position	60	0,50	102

Tabel 2 Calibration result of the second potentiometer

Arm 2			
	Angle (Degree)	**VPout (Volt)**	**ADC Value (Decimal)**
Minimum Position	-60	2,30	472
Center Position	0	1,42	291
Maximum Position	60	0,59	120

Tabel 3 Calibration result of the third potentiometer

Arm 3			
	Angle (Degree)	**VPout (Volt)**	**ADC Value (Decimal)**
Minimum Position	-60	2,33	478
Center Position	0	1,46	300
Maximum Position	60	0,56	115

B. Internal Control Lift Movement Result without Load

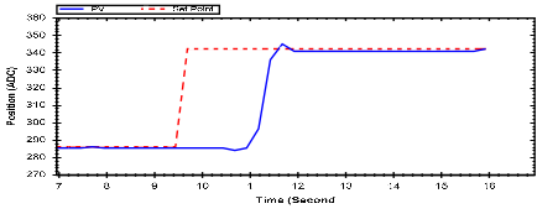


Fig. 7 Response System of the first arm / motor with lift movement task withoud load with RMAC-PID

Fig. 7 shows that from RMAC-PID response, rise time was not too fast because of the damping nature from derivative control that working together with RMAC that has velocity and acceleration control, so that system attempted to damp and to decrease acceleration and velocity so that overshoot would not be occured. This was proven by a very little overshoot on RMAC-PID response. Moreover, oscillations were minimum because on RMAC, the system attempted to zero the acceleration. That minimum oscillations had also been supported by the damping nature of derivative control [16]. On the end of the experiment, we got the errors of 0,042º on first arm, 0,398º on second arm, and 0,447º third arm.

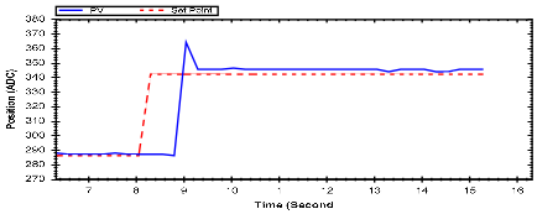


Fig. 8 Response System of the first arm / motor with lift movement task withoud load with RMAC

On RMAC without PID (Fig. 8), rise time was faster because the damping nature of derivative control was not present, resulting in the maximum performance of velocity control from RMAC to reach set point as fast as possible. But, this was resulting in a big overshoot. Moreover, the error elimination from integral control was not present. The oscillations were also bigger than RMAC-PID because the derivative control was not present. At the end of the experiment, we got the of 0,984º on first arm, 0,595º on second arm, and 0,227º on third arm.

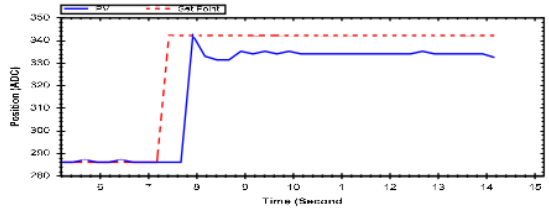


Fig. 9 Response System of the first arm / motor with lift movement task withoud load with PID

On PID (Fig. 9), rise time was similar with RMAC and faster than RMAC-PID. On all arms, the system with PID was failed to reach the set point. In theory, PID should be able to reach the set point. But on the real condition, the robot itself has its own mechanical load that cannot be handled by PID. As the opposite, RMAC can handle this because of its nature to damp outside and inside disturbances. Because the system could not reach the set point, the system was always attempting to reach the set point, resulting in big oscillations. At the end of the experiment, we got the errors of 3,098º on the first arm, 2,715º on the second arm, and 1,458º on the third arm.

C. Internal Control Lift Movement Result with Weighted Load

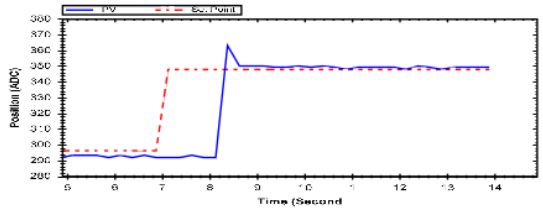


Fig. 10 Response System of the first arm / motor with lift movement task with weighted load with RMAC-PID

The weight used was a 250 gr plier. Shown by Fig. 10 that experiment with weighted load with RMAC-PID was resulting in a bigger overshoot from experiment without load. This was because the velocity control from RMAC attempted to reach set point as fast as possible. Because there was a downward force from the weighted load, the velocity was damped. It was resulting in the increase of the velocity to match the downward force. That faster velocity was increasing the possibility to overshoot higher.

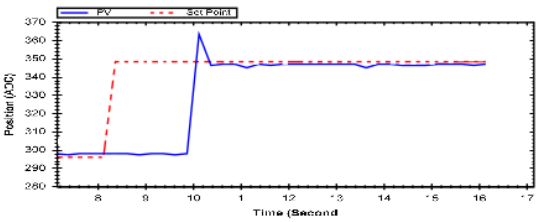


Fig. 11 Response System of the first arm / motor with lift movement task with weighted load with RMAC

On RMAC (Fig. 11), the performance was slightly lower than RMAC-PID. Overshoot and error was slightly bigger than RMAC-PID. This was because the PID was not much help to handle a weighted load disturbance. This was proven from the PID response system from Fig. 12, where the system could not reach the set point. Even so, integral control was slightly reducing the overshoot and error.

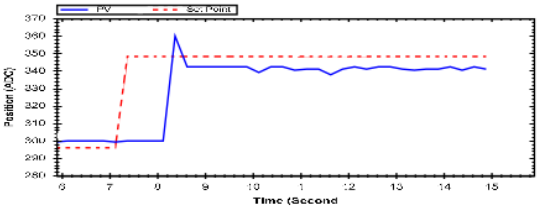


Fig. 12 Response System of the first arm / motor with lift movement task with weighted load with PID

D. Internal-External Control Result for Still Object

Table 4 Still Object Result

No	Result	Time (s)
1	Success	2,0
2	Success	1,9
3	Success	1,7
4	Success	0,9
5	Fail	-
6	Success	1,3
7	Success	1,3
8	Success	1,3
9	Success	7,4
10	Success	1,4

From the manipulation result in tracking and catching on the still object (Table 4), the robot has succeeded to manipulate object at nine times from ten times, or having a success rate of 90%. On the average, the robot was able to catch the object in 2,13 seconds. There was still the failure, because the robot was doing the catching move while the object was not patched with end-effector. That was because of the difference between the set height of the end-effector with the height of the object.

E. *Internal-External Control Result for Non-Linear Movement Object*

Table 5 Non-Linear Object Result

No	Result	Time (s)
1	Success	1,3
2	Success	1,3
3	Success	0,9
4	Success	2,9
5	Fail	-
6	Fail	-
7	Success	1,4
8	Success	2,1
9	Success	3,5
10	Success	1,8

From the manipulation result in tracking dan catching on the non-linear movement object (Table 5), the robot has succeeded to manipulate object at eight times from ten times, or having a success rate of 80%. On the average, the robot was able to catch the object in 1,9 seconds. There was still the failure, because of the similar problems from the experiment with still object. The variability from recorded time was depending on where the object was rolled. The closer the position of rolled ball with the end-effector position, the faster the time needed by the robot to catch the object.

IV. Conclusion and Future Work

In this paper, the design and the implementation of internal-external control for the Delta robot prototype to manipulate non-linear movement object is present. The results show that the designed internal control was proved to be responsive with 1,17 seconds rise time average and accurate with 0,289º error average on all arms / motors. The combination of the internal and external control was able to manipulate the still object with 90% success rate and 2,13 seconds time average. The non-linear movement object was able to be manipulated with 80% success rate and 1,9 seconds time average. The outcome of this experiment possible the low cost Delta robot that fast and accurate, because the use of the ultra accurate structure and motor torque are not mandatory. Also, this method will reduce the use of conveyor because the manipulated object is not needed to move linearly.

The next work for the development of this research is the use of rotary encoder with Kalman filter to replace the use of potentiometer. Then we will use gain scheduling with Fuzzy logic to tune the parameter values from RMAC-PID and PID inside the internal-external control. The design of the Delta robot prototype will also be improved. The use of the torque control will also be used with the use of higher torque motors.

Designing Internal-External Control Method forDelta Robot Prototype to Manipulate Non-Linear MovementObject

ORIGINALITY REPORT

2%	2%	0%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	icitacee.undip.ac.id Internet Source	2%

Exclude quotes	On	Exclude matches	< 15 words
Exclude bibliography	On		

Designing Internal-External Control Method forDelta Robot Prototype to Manipulate Non-Linear MovementObject

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6