Using MATLAB to Animate the Generation of a Space Centrode in Kinematics

Glynn P. Adams, Ing-Chang Jong
University of Arkansas

Abstract

Computer animations are valuable tools for demonstrating and teaching dynamics of mechanisms. Several software packages are commercially available to aid in this effort. The work presented in this paper is intended to complement, the animation capabilities of existing software by using the **MATLAB** programming language to animate a specific four-bar linkage and display the space centrode of the coupler during the animation. The linkage presented here is a crank-rocker mechanism, which can be assembled in a colinear configuration. This linkage was selected because of the interesting nature of the coupler link space centrode and the motion of the output, link. The position solution for the linkage is obtained with a. Newton-Ra.phson method and the use of kinematic coefficients. The details of this approach are presented as is the specific **MATLAB code** required to produce the position solution and the animation.

Introduction

One of the main impediments to learning dynamics of mechanisms is the visualization of the mechanism motion. Several commercially available software packages such as Working Model and *Analytix* allow users to create mechanisms with various constraints and produce animations of the resulting motion. The packages provide a valuable asset for helping students to realize the kinematics associated with a wide variety of mechanisms. However, computer programs written for specific mechanisms and specific purposes can serve as excellent complements to these software packages The animation of a specific mechanism involves both computer programming and numerical methods. The objective of this paper is to share the programming strategy with instructors who may contemplate animating a linkage and the generation of a space centrode using **MATLAB.**

Consider the four-bar linkage in Fig. 1, which can be assembled in a collinear configuration. For example, we can choose r_1 = 0.3 m, r_2 = 0.5 m, r_3 = *1.0* m, and r_4 = *1.2* m. The crank *AB,* coupler BD, and output DE all become collinear along the ground link when $\theta_1 = \pi$. Animation of this mechanism and generation of the space centrode of the coupler are particularly interesting for the following reasons:

- The crank must make two revolutions to completely define the motion of the output.
- The space centrode of the coupler contains two asymptotes.

The analytical study of the space centrode and its asymptotes was presented by Jong, et al[1].

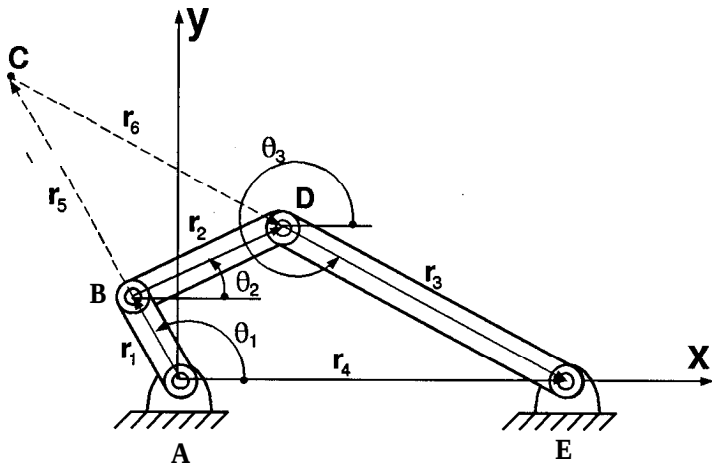


Figure 1: Four-bar linkage

Position Solution and Kinematic Coefficients

In order to animate a four-bar linkage, the position solution must be obtained for discrete values of the input variable θ_1. A Newton-Raphson method is used here to obtain the position solution. The constraint equations for the mechanism in Fig. 1 are

$$r_1 \cos\theta_1 + r_2 \cos\theta_2 + r_3 \cos\theta_3 - r_4 = 0 \tag{1}$$

$$r_1 \sin\theta_1 + r_2 \sin\theta_2 + r_3 \sin\theta_3 = 0 \tag{2}$$

Hall[2] suggests that the computation of the position solution can be enhanced by the use of kinematic coefficients. A kinematic coefficient is defined as the derivative of a dependent variable with respect to an independent variable in kinematics. In the present case, θ_1 is the independent variable, while θ_2 and θ_3 are the dependent variables. Differentiating Eqs. (1) and (2) with respect to θ_1, we get

$$-r_1 \sin\theta_1 - (r_2 \sin\theta_2)h_2 - (r_3 \sin 193)h3 = 0 \tag{3}$$

$$r_1 \cos\theta_1 + (r_2 \cos\theta_2)h_2 + (r_3 \cos\theta_3)h_3 = 0 \tag{4}$$

where h_2 and h_3 are the kinematic coefficients. Equations (3) and (4) yield

$$h_2 = \frac{d\theta_2}{d\theta_1} = \frac{r_1 \sin(\theta_3 - \theta_1)}{r_2 \sin(\theta_2 - \theta_3)} \tag{5}$$

$$h_3 = \frac{d\theta_3}{d\theta_1} = \frac{r_1 \sin(\theta_1 - \theta_2)}{r_3 \sin(\theta_2 - \theta_3)} \tag{6}$$

From Eqs. (5) and (6), we see that $d\theta_2 = h_2 d\theta_1$ and $d\theta_3 = h_3 d\theta_1$, or $\Delta\theta_2 = h_2 \Delta\theta_1$ and $\Delta\theta_3 = h_3 \Delta\theta_1$. When we do the iterative solution (e.g., Newton-Raphson method), we typically use the values of the dependent variables from the previous position as the starting

estimates. According to Hall', the iterative process can be improved in a simple way by using the kinematic coefficients as follows:

$$\theta_2(i+1) = \theta_2(i) + h_2\Delta\theta_1 \tag{7}$$
$$\theta_3(i+1) = \theta_3(i) + h_3\Delta\theta_1 \tag{8}$$

These improved initial estimates for θ_2 and θ_3 are then employed in the Newton-Raphson method. This approach has helped the animation in passing through the collinear configuration, thereby displaying the complete motion of the output link DE.
Furthermore, the location C (:x~., yC) of the velocity center can be expressed in terms of the kinematic coefficients. Referring to Fig. 1, we write

$$\mathbf{r}_c = \mathbf{r}_1 + \mathbf{r}_5 \tag{9}$$

An additional vector loop involving the velocity center can be written as

$$\mathbf{r}_5 + \mathbf{r}_6 = \mathbf{r}_2 \tag{10}$$

The x and y components of Eq. *(10)* can be expressed in matrix form as

$$\begin{bmatrix} \cos\theta_1 & \cos\theta_3 \\ \sin\theta_1 & \sin\theta_3 \end{bmatrix} \begin{Bmatrix} r5 \\ r6 \end{Bmatrix} = \begin{Bmatrix} r2\cos\theta_2 \\ r2\sin\theta_2 \end{Bmatrix} \tag{11}$$

Solving Ey. (11) for the magnitude of $\mathbf{r}_5$ and employing trigonometric identities, we get

$$r_5 = \frac{r_2\sin(\theta_3 - \theta_2)}{\sin(\theta_3 - \theta_1)} \tag{12}$$

From Eqs. (5) and (12), it can be shown that

$$r_5 = -\frac{r_1}{h_2} \tag{13}$$

Therefore, the coordinates (x_c, y_c) of the velocity center Gin Fig. 1, as given by Eq. (9), ase

$$x_c = \left(r_1 - \frac{r_1}{h_2}\right)\cos\theta_1 \tag{14}$$
$$y_c = \left(r_1 - \frac{r_1}{h_2}\right)\sin\theta_1 \tag{15}$$

Description of the Animation

The animation as well as programming described here is implemented using MATLAB *Version 4.0.* Two separate programs are included. One program is used to compute and store the position solution and velocity center for discrete values of θ_1. The other program uses the stored data to animate the mechanism. A listing of these two programs is provided, where comments describing the functions of each segment are included.

The program "position.m" implements the steps described above to obtain the position solution of the mechanism. The position solution is computed for two complete revolutions of the crank. The program "animate.m" produces an animation of the mechanism as well as the generation of the space centrode of the coupler. The values of the animated angular positions are displayed at the bottom of the screen. The instantaneous velocity center of the coupler is displayed as a single dot marker. These markers remain during the animation to show the locus of the velocity center. Once the crank has completed two revolutions, the complete space centrode is traced by connecting the velocity center markers with lines. The tracing allows the viewer to more easily follow the generation of the space centrode.

The animation is accomplished using MATLAB features known as graphics objects and graphics handles. Graphics objects are the basic drawing primitives used to create images. The main graphics objects used here are

- figure
- plot
- text

Graphics handles are unique identifiers assigned to the objects. These handles allow the user to access the various properties associated with each graphic object. The following handles are defined in the program.

- screen: This is the handle to the figure in which the animation appears.
- crank, coupler, *out*, *pinl*, and pin2: These are handles to the mechanism components.
- *instcen*, *AC*, and EC: These are handles to the velocity center and lines drawn to locate it.
- t1val, t2val and t3val: These are handles to the text displays of position values.

Different properties can be specified for each of the graphics objects. For example, a *plot* object has the properties of 'Xdata,' 'Ydata,' 'Color,' and 'Linewidth,' among others. Once a graphics object is created, its properties can be changed within the program using the *set* command and the object's handle. For instance, the crank is initially drawn at $\theta_1 = 0$. Its position is changed with a *set* command that resets the properties 'Xdata' and 'Ydata' for the *plot* identified by the handle crank. The remaining components are repositioned in the same manner.

Additional MATLAB commands are used in generating the animation. They are

- *hold on:* This allows each *plot* command to be displayed simultaneously.
- axis: This sets the limits of the plots produced in the figure.
- *draw now:* This empties the graphics buffer and updates the figure to represent the current status of all graphics objects.

MATLAB Program

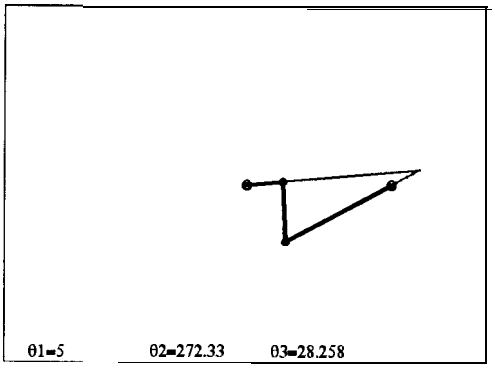
The following section contains the code produced to generate the animation using MATLAB. The file Position.m is used to compute the position solution of the mechanism for increments of the crank angle θ_1. The file Animate.m is used to produce the animation and save the displays to graphics files.

```
%------------- POSITION.M  by GLYNN P. ADAMS   DECEMBER 1996  ---------%
  clear; dtor = pi/180.;        % convert degrees to radians
  r1=0.3; r2=0.5; r3=1.0; r4=1.2;      % The link lengths
  tol = 1.0E-5;                        % Tolerance for Newton-Raphson
%------------- Crank angle -> initial, increment and final       ---------%
  theta1 = 0*dtor; incr1  = 1.00*dtor; final1 = theta1 + 720*dtor;
%------------- Initial estimates of theta2 and theta3     ---------%
  theta2 = 270*dtor;  theta3 = 20*dtor;
  t = [theta1,theta2, theta3];          % Working array for theta estimates
  for i = 0: ceil(final1/incr1);        % 'for loop' for crank increments
    t(1) = theta1 +(i*incr1);
    if(   ( 0.99*pi < t(1) & t(1) < l.Ol*pi ) . . . .
        |( 0.97*3*pi < t(1) & t(1)< 1.03*3*pi ) );
      t(1) = theta1 +(i*incr1)+(0.05*dtor);      end;
    dtheta = [2*tol;2*tol]; iters = 0;
%------------- 'While Loop' for Newton-Raphson routine    ---------%
  while ( ( abs(dtheta(1)) > tol| abs(dtheta(2)) > tol ) & (iters < 50 ) )
    iters = iters + 1;
    eps1 = r1*cos(t(1)) + r2*cos(t(2)) + r3*cos(t(3)) -r4;
    eps2 = r1*sin(t(1)) + r2*sin(t(2)) + r3*sin(t(3));
    A = [-r2*sin(t(2)) -r3*sin(t(3)); r2*cos(t(2)) r3*cos(t(3))];
    B = [-eps1; -eps2];     dtheta = A\B;    t(2:3) = t(2:3) + dtheta'; end;
%------------- End of Newton-Raphson routine for this crank angle ---------%
%------------- Save the position solution for later animation     ---------%
   if(i == 0) angles=t; else angles = [angles;t]; end;
%------------- Compute kinematic coeff to improving next estimate ---------%
   hvec = A\[r1*sin(t(1));-r1*cos(t(1))];      t(2:3) = t(2:3) + hvec'*incr1;
%------------- Compute coupler velocity center for position       ---------%
   if(i==0) VC = [(r1-r1/hvec(1))*[cos(t(1)),sin(t(1))]]; else;
     VC = [VC;(r1-r1/hvec(1))*[cos(t(1)),sin(t(1))]]; end; end;
%------------- End of the 'for loop' for crank angles increments  ---------%
  animate
%------------- (The remaining code begins a new file)             ---------%
%------------- ANIMATE.M  by   GLYNN P. ADAMS   DECEMBER 1996   ---------%
  for i=1:length(angles);
    if(angles(i,2) < 0);  angles(i,2) = angles(i,2)+2*pi; end;
    if(angles(i,3) < 0);  angles(i,3) = angles(i,3)+2*pi; end;
    angles(:,2:3) = rem(angles(:,2:3),2*pi);   end;
%------------- Initialize x and y arrays for animation            ---------%
  x1 = [zeros(length(angles),1)]; x2 = [r1*cos(angles(:,1))];
  x3 = [r1*cos(angles(:,1)) + r2*cos(angles(:,2))];
  x4  = [r4*ones(length(angles),1)]; x = [x1,x2,x3,x4];
  y1 = [zeros(length(angles),1)]; y2 = [r1*sin(angles(:,1))];
  y3 = [r1*sin(angles(:,1)) + r2*sin(angles(:,2))];
  y4 = [zeros(length(angles),1)]; y = [y1,y2,y3,y4];     plotit = 0;
  screen = figure;   % Open an animation window, with handle 'screen'
  posit = [2.0 2.0 10.0 8.01;   set(screen,'Position',posit,'Color',[0 0 0]);
  set(screen,'Units','inches','NumberTitle','off','Name','4-Bar  Animation');
  hold on;             % Hold the current images when new images are drawn
%------------- Plot the anchors for the crank and the output      ---------%
  plot(0,0,'w*','markersize',4); plot(0,0,'wo','markersize',8);
  plot(r4,0,'w*','markersize',4); plot(r4,0,'wo','markersize',8);
%------------- Set the axis dimensions and remove the axis frame  ---------%
  axis([-2.0 2.0 -1.5 1.5]);     axis('equal');       axis('off');
  border=plot([-2.0 2.0 2.0 -2.0 -2.0],[-1.5 -1.5 1.5 1.5 -1.5],'m'); % Border
%------------- Draw the mechanism in the initial configuration    ---------%
  instcen = plot(VC(1,1),VC(1,2),'g.');  crank = plot(x(1,1:2),y(1,1:2),'r','linewidth',3.0);
  out = plot(x(1,3:4),y(1,3:4),'b','linewidth',3.0);  coupler = plot(x(1,2:3),y(1,2:3),'w','linewidth',3.0);
```

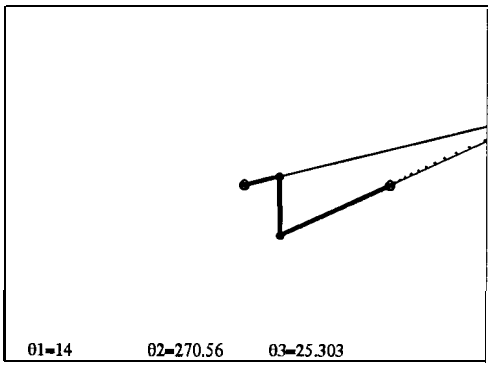
```
  pin1 = plot(x(1,2),y(1,2),'wo','markersize',6.0); pin2 = plot(x(1,3),y(1,3),'wo','markersize',6.0);
%-------------  Draw lines to locate the velocity center     ---------%
  AC = plot([x(1,2),VC(1,1)],[y(1,2),VC(1,2)],'w'); EC = plot([x(1,3),VC(1,1)],[y(1,3),VC(1,2)],'w');
%-------------  Display the current angles                  ---------%
  t1str =[num2str(angles(1,1)/dtor,5)]; t2str =[num2str(angles(1,2)/dtor,5)];
  t3str= [num2str(angles(1,3)/dtor,5)];
  t1val = text(-1.8,-1.4,['q1=',num2str(angles(1,1)/dtor,5)],.....
           'fontname','symbol','fontsize',16,'Color',[1 1 1]);
  t2val = text(-0.8,-1.4, ['q2=',num2str(angles(1,2)/dtor,5)],.....
           'fontname','symbol','fontsize',16,'Color',[1 1 1]);
  t3val = text( 0.2,-1.4, ['q3=',num2str(angles(1,3)/dtor,5)],.....
           'fontname','symbol','fontsize',16,'Color',[1 1 1]); drawnow;
%-------------  Define an angle for generating a still frame      ---------%
  plotme = 30.0; frame = (plotme-angles(1,1))/(incr1/dtor)+1;
  step = 10;  jump = step/(incr1/dtor); % Animation increment angle
  for kk=1+jump:jump:length(angles);        i=kk;
    if( (kk < frame) & (frame< (kk+jump) )| frame==kk );
      i=frame; plotit=1; end;
    set(instcen,'Xdata',VC(1:i,1),'Ydata',VC(1:i,2)); set(crank,'Xdata',x(i,1:2),'Ydata',y(i,1:2));
    set(out,'Xdata',x(i,3:4),'Ydata',y(i,3:4)); set(coupler,'Xdata',x(i,2:3),'Ydata',y(i,2:3));
    set(pin1,'Xdata',x(i,2),'Ydata',y(i,2)); set(pin2,'Xdata',x(i,3),'Ydata',y(i,3));
    set(AC,'Xdata',[x(i,2),VC(i,1)],'Ydata',[y(i,2),VC(i,2)]);
    set(EC,'Xdata',[x(i,3),VC(i,1)],'Ydata',[y(i,3),VC(i,2)]);
    set(t1val,'string',['q1=',num2str(angles(i,1)/dtor,5)]);
    set(t2val,'string',['q2=',num2str(angles(i,2)/dtor,5)]);
    set(t3val,'string',['q3=',num2str(angles(i,3)/dtor,5)]);
    drawnow;
    if(plotit == 1); filename=['crank',num2str(plotme,4)];
      eval(['print -deps ',filename,'.ps']); plotit=0;
      disp(['DUMPING THIS PLOT TO ',filename,'.ps']);
    end;
  end;
  if(i ~= length(angles)); % Include last position at 720 degrees
    i = length(angles);
    set(instcen,'Xdata',VC(1:i,1),'Ydata',VC(1:i,2));
    set(crank,'Xdata',x(i,1:2),'Ydata',y(i,1:2));
    set(out,'Xdata',x(i,3:4),'Ydata',y(i,3:4));
    set(coupler,'Xdata',x(i,2:3),'Ydata',y(i,2:3));
    set(pin1,'Xdata',x(i,2),'Ydata',y(i,2));
    set(pin2,'Xdata',x(i,3),'Ydata',y(i,3));
    set(AC,'Xdata',[x(i,2),VC(i,1)],'Ydata',[y(i,2),VC(i,2)]);
    set(EC,'Xdata',[x(i,3),VC(i,1)],'Ydata',[y(i,3),VC(i,2)]);
    set(t1val,'string',['q1=',num2str(angles(i,1)/dtor,5)]);
    set(t2val,'string',['q2=',num2str(angles(i,2)/dtor,5)]);
    set(t3val,'string',['q3=',num2str(angles(i,3)/dtor,5)]);
    drawnow;      end;
  set(AC,'visible','off'); set(EC,'visible','off');
  plot([-3.5,3.5],[0,0],'y');
  for k=2:length(angles);   % Trace the coupler space centrode
    j=max(1,k-5);
    plot(VC(j:k,1),VC(j:k,2),'w','erasemode','none'); pause(0.2);  end;
```

Output of the MATLAB Program: Sample Frames in the Animation

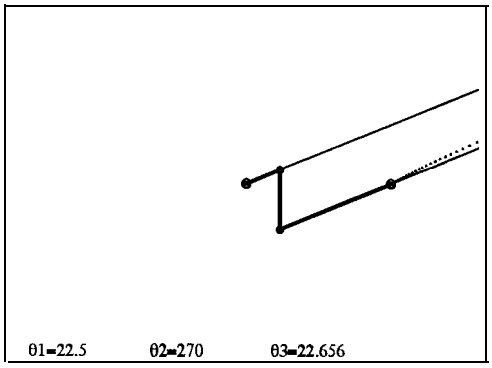
The frames shown here are generated using the print command in MATLAB. The variables plotme and *frame* are used in the file *Animate.m* to set the crank angle at which a frame is generated upon execution of the program.



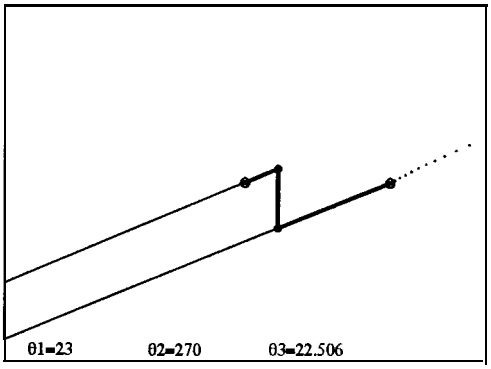
(a) θ_1 = **5"**



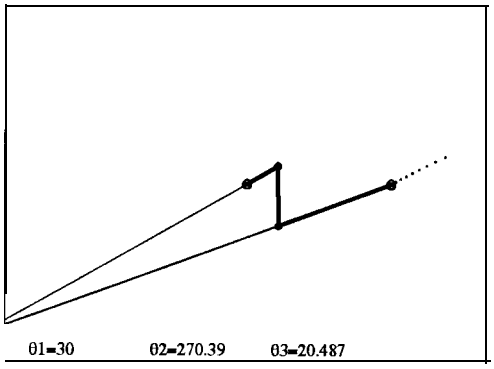
(b) θ_1 = **14"**



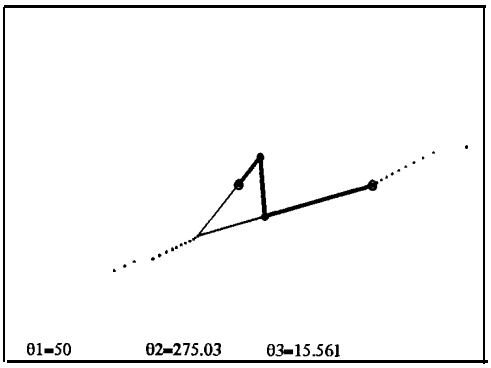
(c) θ_1 = **22.5"**



(d) θ_1 = **23"**

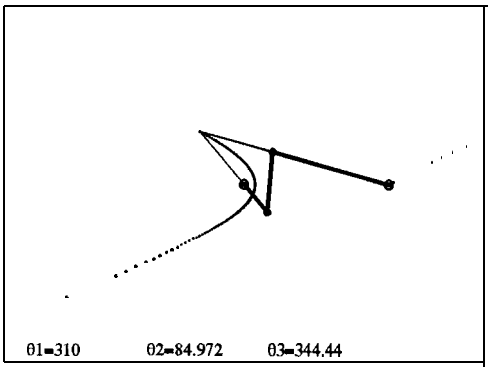


(e) θ_1 **= 30"**

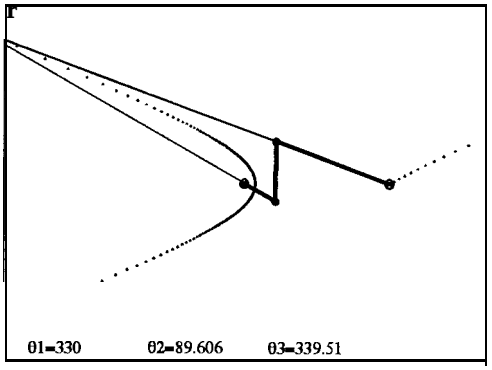


(f) θ_1 = **50"**

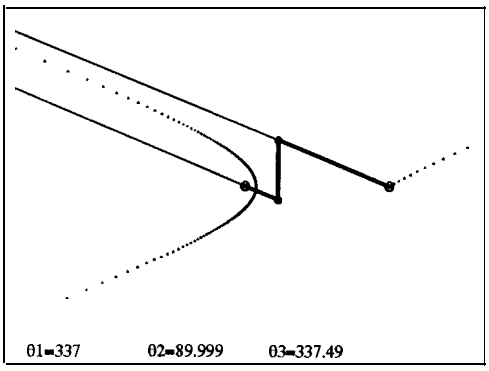
Sample Frames in the Animation (continued)



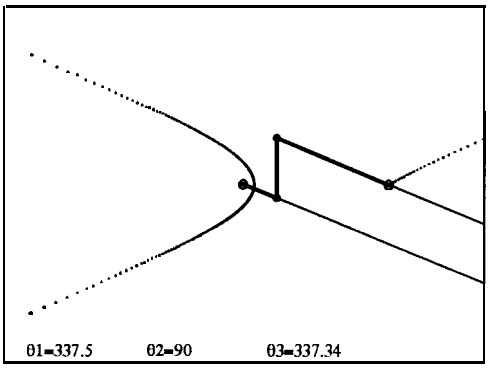
(g) θ_1 = **3 1 0 ”**



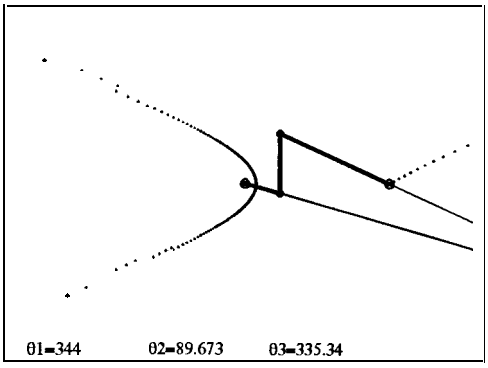
(h) θ_1 **= 330”**



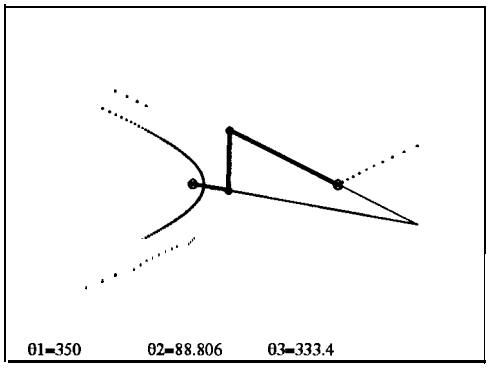
(i) θ_1 **= 337”**



(j) θ_1 **= 337.5”**

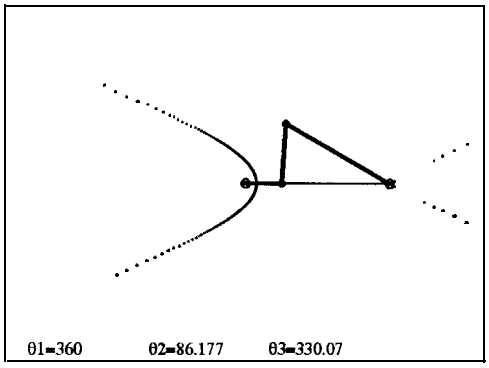


(k) θ_1 **= 344”**

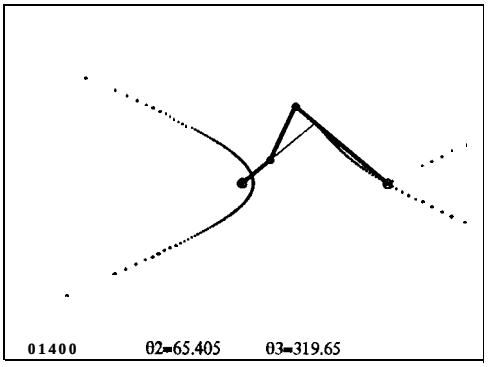


(1) θ_1 **= 350”**

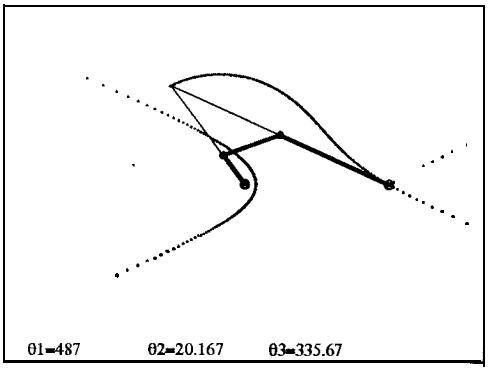
Sample Frames in the Animation (continued)



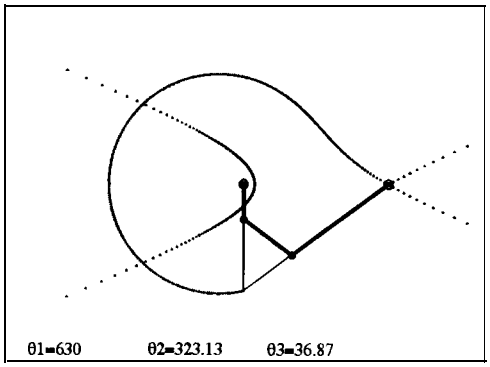
(m) $\theta_1 = 360$"



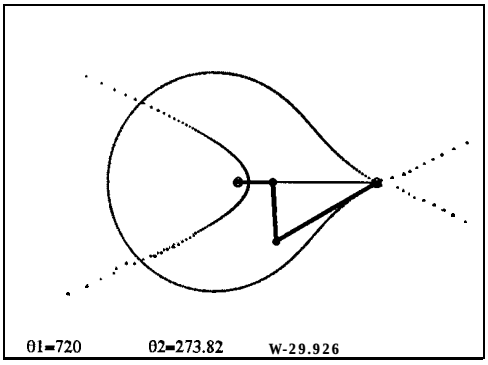
(n) $\theta_1 = 400$"



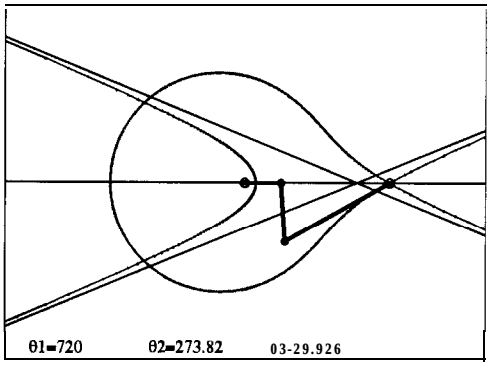
(o) $\theta_1 = 487$"



(p) $\theta_1 = 630$"



(q) $\theta_1 = 720$"



(r) $\theta_1 = 720$" with asymptotes

Concluding Remarks

It should be noted that a movie command is also available in MATLAB for animation purposes. To use this command, a series of figures which represent individual frames in a. sequence must be generated, stored in memory and displayed sequentially. Due to possible memory limitations, the current, method was chosen over the use of the movie command. Other software packages, such as QuickBASIC[3] and MATHEMATICA,[4] can be employed to complement existing animation software on the market. Nevertheless, MATLAB is increasingly available to both instructors and students. This paper is written as part of an overall effort to provide multimedia tools for modern teaching.

The use of kinematic coefficients in modifying the iterative Newton-Raphson method to obtain the position solution has been briefly discussed here. The advantages of this approach can be seen by modifying the computer code to ***comment out*** the line "t(2:3) = t(2:3)+ hvec'*incr1," appearing after the end of the Newton- Raphson routine. If the kinematic coefficients are not employed, the position solution generated by this program may not pass through the collinear configuration. The four-bar linkage is often used in studying the kinematics of connected rigid-body motions. Often the concepts of velocity centers and space centrodes are difficult for students to grasp. The inclusion of singularities in the linkage kinematics only complicates the matter. Through mechanism animations these concepts can be demonstrated in a, manner that gives the students a more intuitive feel for the physical significance of each of these concepts.

References

1. Jong, I. C., Reynolds, R. R., and Adams, G. P., "Determination of Space Centrode of a Coupler Link," Session *2668, 1996 ASEE* ***Annual Conference Proceedings*** Washington, D.C., June 23-26.

2. Hall, A. S. Jr., Notes on. Mechanism *Analysis,* Waveland Press, Inc., Prospect Heights, IL, 1986.

3. Jong, I. C., and Onggowijaya, S. N., "Animation Programming with QuickBASIC to Aid the Teaching of Kinenmtics," to be published.

4. Reynolds, R. R., and Jong, I. C., "Using MATHEMATICA to Animate the Generation of a Space Centrode in Kinematics," to be published.

Glynn P. Adams

Glynn Adams received BSME and MSME degrees from Louisiana State University and a. Ph.D from Purdue University He is an Assistant Professor of Mechanical Engineering at the University of Arkansas studying mechanics and mechanical design. He served as a NASA/ASEE Summer Faculty Fellow during 1996 and 1997.

Ing-Chang Jong

Ing-Chang Jong received a. BSCE from the National Taiwan University in 1961, an MSCE from the SDSM&T in 1963, and a Ph.D. from Northwestern University in 196.5. He is a Professor of Mechanical Engineering at the University of Arkansas. He and Dr. Bruce G. Rogers published an engineering mechanics textbook in 1991. He is serving as the Chair of the Mechanics Division, 1996-1997.