Int. J. Engng Ed. Vol. 13, No. 6, p. 417–425, 1997
Printed in Great Britain.

0949-149X/91 $3.00+0.00

Dimensional Optimization for the Crank-Rocker Mechanism using TK Solver

K. LIU AND W. P. BOYLE
Division of Engineering, Saint Mary's University, Halifax, Nova Scotia, Canada, B3H 3C3

Using the computational package TK Solver, optimal position and velocity syntheses are performed for the crank-rocker mechanism. The performance measure for optimization is defined as the least squares error between the desired motion and the actual motion, with relations between the design variables and the performance measure being derived from vector diagrams. An optimization for the dimensions of two links is carried out by a one dimensional search, in combination with the list solving facility provided by TK. This paper shows that this type of mechanism synthesis problem is easily solved by the application of numerical optimization, and the convenient computational features of the TK Solver package.

SUMMARY OF EDUCATIONAL ASPECTS OF THIS PAPER

1. The paper discusses materials for a course in Dynamics of Rigid Bodies.
2. Students of all branches of Engineering are taught in this course.
3. Level of the course: Year 2.
4. Mode of presentation is lecture/tutorial.
5. The material is presented in a regular course.
6. The hours required to cover the material is $3 + 3$ per week.
7. Student homework or revision hours required for the materials is about 4.
8. The paper presents an 'easy' method to address the 'difficult' problem of linkage length optimization.
9. The standard text recommended in the course, in addition to author's notes is Bedford and Fowler: *Dynamics*.
10. The paper describes software applications useful in the Mechanical Engineering disciplines for Years 2 onwards using TK Solver library search 'Golden' employed for linkage optimization. The programs allow an 'infinite' variety of exercises to be set for students which have been tested in the classroom or in project work in a very limited fashion. This paper extends some of the elements of previous papers by the authors using the concept of pseudographics and TK Solver as the working tool. Programming in a formal language is replaced by TK code and TK library functions—a big convenience.

* Accepted 15 March 1996.

INTRODUCTION

IN courses such as Dynamics of Machines, linkage synthesis is an important topic. Traditionally two methods are used

- graphical synthesis
- analytical synthesis.

The graphical approach is intuitive, but the method can be tedious and inaccurate when a multi-position solution is required. The analytical method uses successive differentiation of the position vector equation to yield expressions for velocity and acceleration. The algebraic nature of this procedure makes it suitable for the application of computers, but formulating the relations between the design parameters and the desired motion path can be demanding. Without an explicit relation, computation is difficult to perform.

Norton [1] uses complex number algebra to execute an analytical solution. However, the mathematics involved are somewhat tortuous, and may become discouraging to students who are not facile with complex number manipulation.

The graphical solution depends essentially on a series of vector diagrams, and Boyle [2] has devised a computational/graphical method, named 'pseudographics', to generate Cartesian co-ordinates for the vector diagrams. Rather than depending on vector algebra, the method has a computational foundation, and preserves the intuitive features of the graphical solution. However, pseudographics result in a set of relations in which it is not easy to separate the unknowns from the knowns, a necessary first step to programming in a conventional language.

The rule-based and backsolving features of TK Solver [3] make it very suitable for handling the

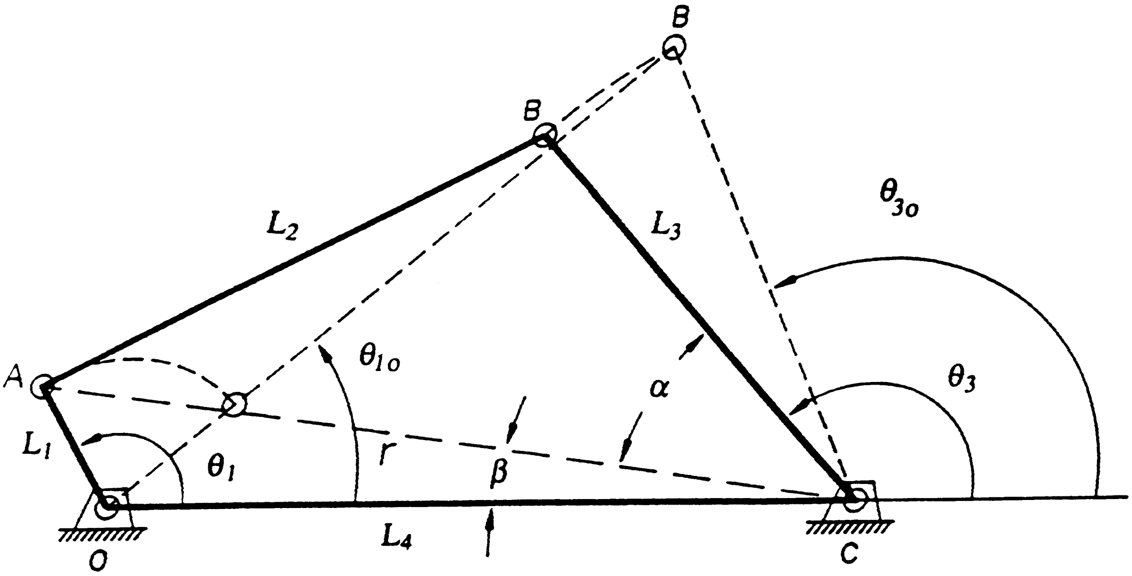


Fig. 1. Crank-rocker mechanism.

relations derived from vector diagrams. Students need only find a sufficient number of rules to allow the model to solve for the unknowns; isolation of unknowns is mostly unnecessary.

If no direct and explicit relation between the desired motion and the design parameters can be found, the design problem can be approached by trial and error methods, typically demanding a high level of computational power. However, when it is possible to specify an exact relation, not necessarily in an explicit fashion, between the design objective and the control variables, then the use of an optimization technique may mean that a high quality design can be obtained

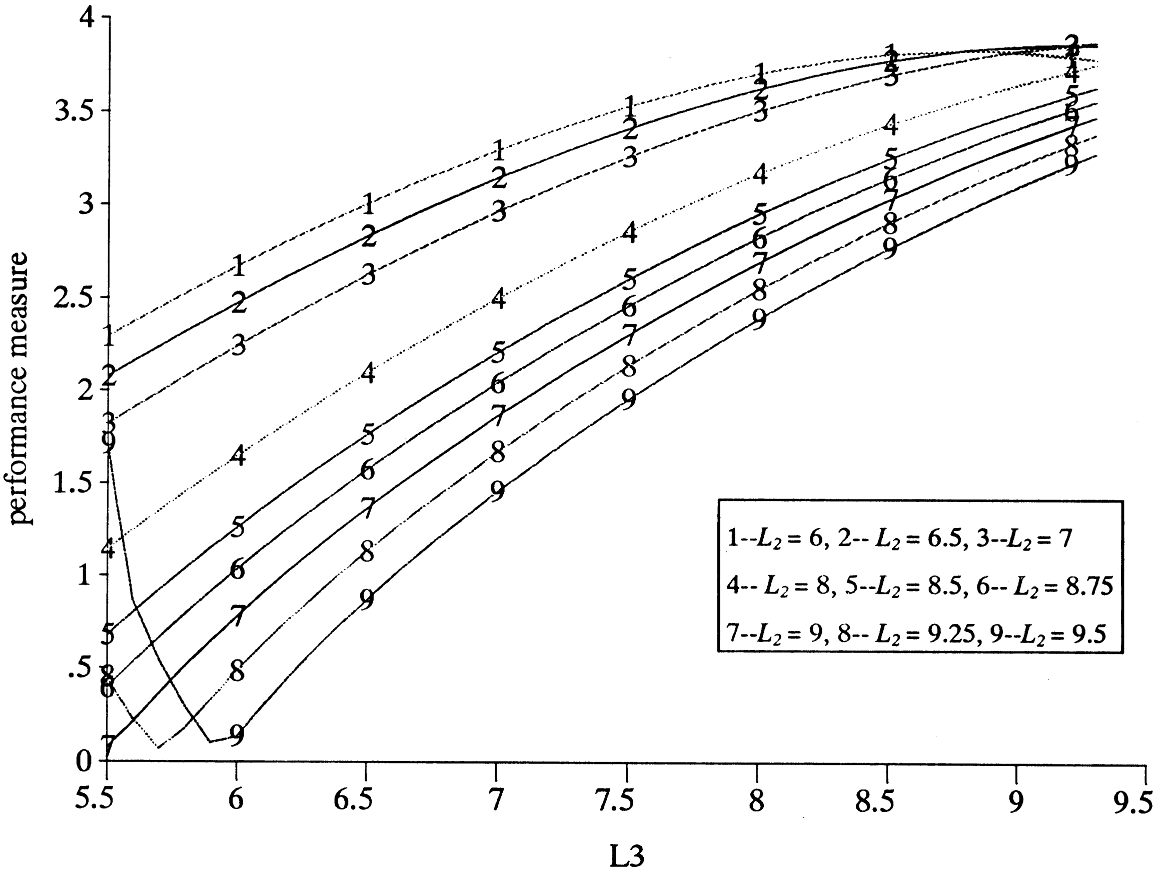


Fig. 2. Plot of performance measure vs. length L_3 for various values of length L_2.

```
.S  Rule
*   L31=L1 + L4 - L2          ; L2+L3 >= L1+L4   Grashof condition
*   L32=L1+L2-L4              ; L3+L4 >= L1+L2   Grashof condition
*   if (L31 > L32) then L3L=L31+.2 else L3L=L32+.2  ; lower limit of L3
*   L3U=(L2+L4-L1)-.2         ; L1+L3 <= L2+L4   Grashof condition, upper limit of L3
*   call Golden(L3L, L3U, 'POSPE; optL3, pospe)  ; Golden search for the optimal L3, position synthesis
```

(a)

```
Comment:                 POSPE for evaluating performance measure
Parameter Variables:     L1,L2,L4,n,m
Input Variables:         L3
Output Variables:        pospe
Statement
theta10=acos(((L1+L2)^2+L4^2-L3^2)/(2*(L1+L2)*L4))
theta30=acos(((L1+L2)^2-L3^2-L4^2)/(2*L3*L4))
pospe=0
for i=1 to n+1
'theta1[i]=theta10+pi()*(i-1)/(2*n)
call POS('theta1[i], L3; 'theta2[i], 'theta3[i])
'theta3d[i]=theta30+2*('theta1[i]-theta10)^m/(3*pi())      ; desired rocker motion
pospe=pospe+('theta3d[i] - 'theta3[i])^2      ; sum performance measure
next i
pospe=sqrt(pospe)/('theta3d[n+1]-'theta3[1])  ; performance measure
call statmsg("L3=",L3,"  pospe=", pospe)
```

(b)

```
Comment:                 POS for evaluating rocker position
Parameter Variables:     L1,L2,L4
Argument Variables:      theta1,L3
Result Variables:        theta2,theta3
Rule
r=sqrt(L1^2+L4^2-2*L1*L4*cos(theta1))
L2^2=r^2+L3^2-2*r*L3*cos(alpha)
L1^2=r^2+L4^2-2*L4*r*cos(beta)
if and (theta1 > 0, theta1 <= pi()) then theta3=pi() - alpha - beta else theta3=pi() - alpha + beta
theta2=atan2((L3*sin(theta3)-L1*sin(theta1)),(L4+L3*cos(theta3)-L1*cos(theta1)))
L2*sin(theta2)=L3*sin(theta3)-L1*sin(theta1)
L2*cos(theta2)=L4+L3*cos(theta3)-L1*cos(theta1)
```

(c)

Fig. 3. TK program for position optimization. (a) Rule Sheet. (b) Procedure Function POSPE. (c) Rule Function POS for position synthesis. (d) Variable Sheet. (e) Table Sheet.

St	Input	Name	Output	Unit	Comment
	1	L1			
L	9.5	L2			
	5	L4			
	30	n			
	1	m			
		L31	-3.5		
		L32	5.5		
		L3U	13.3		
		L3L	5.7		
L		optL3	6.091		
L		pospe	.491		

(d)

Title: results of list-solving for L2

	L2	optL3	pospe
1	4	2.918	.6158
2	4.5	2.894	.5977
3	5	2.969	.5814
4	5.5	3.135	.5669
5	6	3.375	.5539
6	6.5	3.673	.5422
7	7	4.015	.5316
8	7.5	4.391	.522
9	8	4.792	.5132
10	8.5	5.212	.5052
11	9	5.646	.4978
12	9.5	6.091	.491
13	10	6.545	.4848

(e)

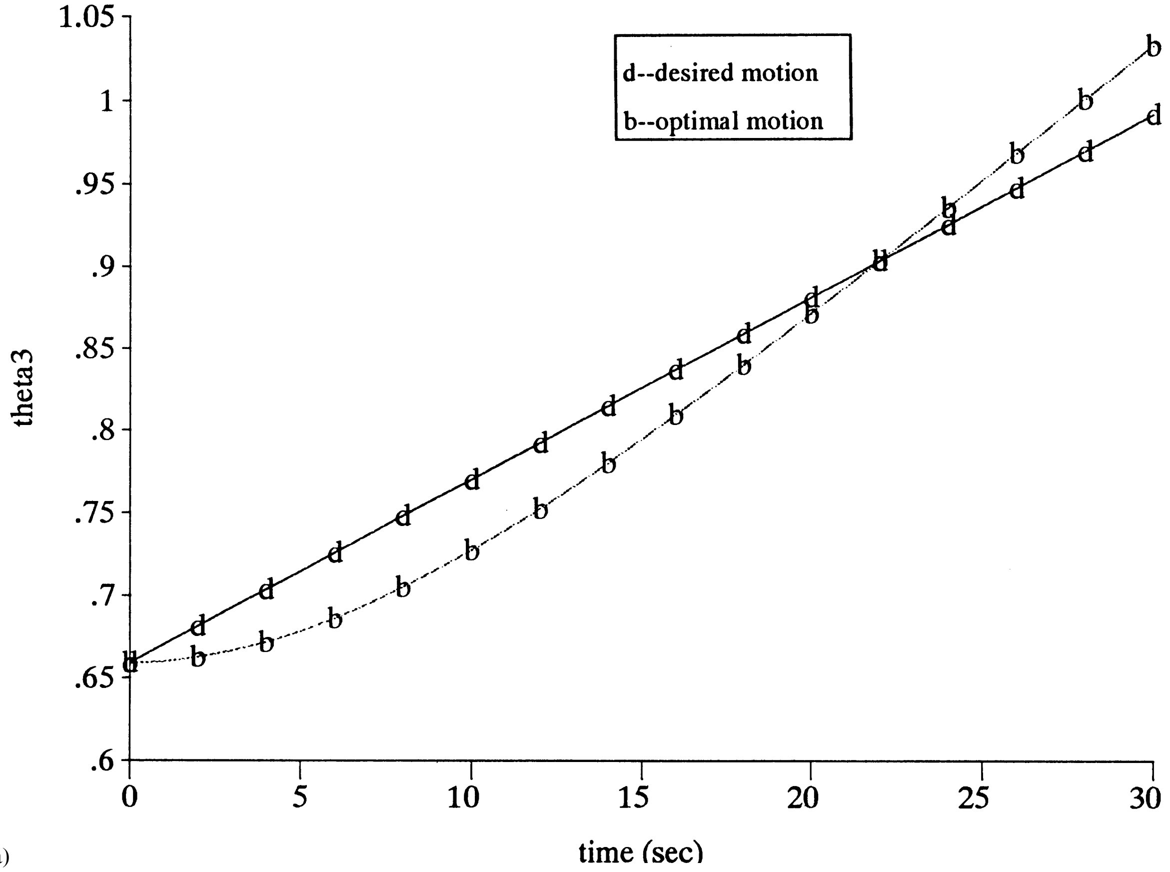
Fig. 3. (Continued).

with efficiency [4]. A linkage synthesis problem can be easily modelled *vis à vis* optimization. The performance measure of a best design can be defined as the least squares error between the desired motion and the actual motion, with parameters such as the lengths of the links chosen in such a way that the performance measure is minimized. TK has a library of optimization algorithms, and in this paper the package is used, in conjunction with pseudo-graphics, to demonstrate linkage optimization for a crank-rocker mechanism.

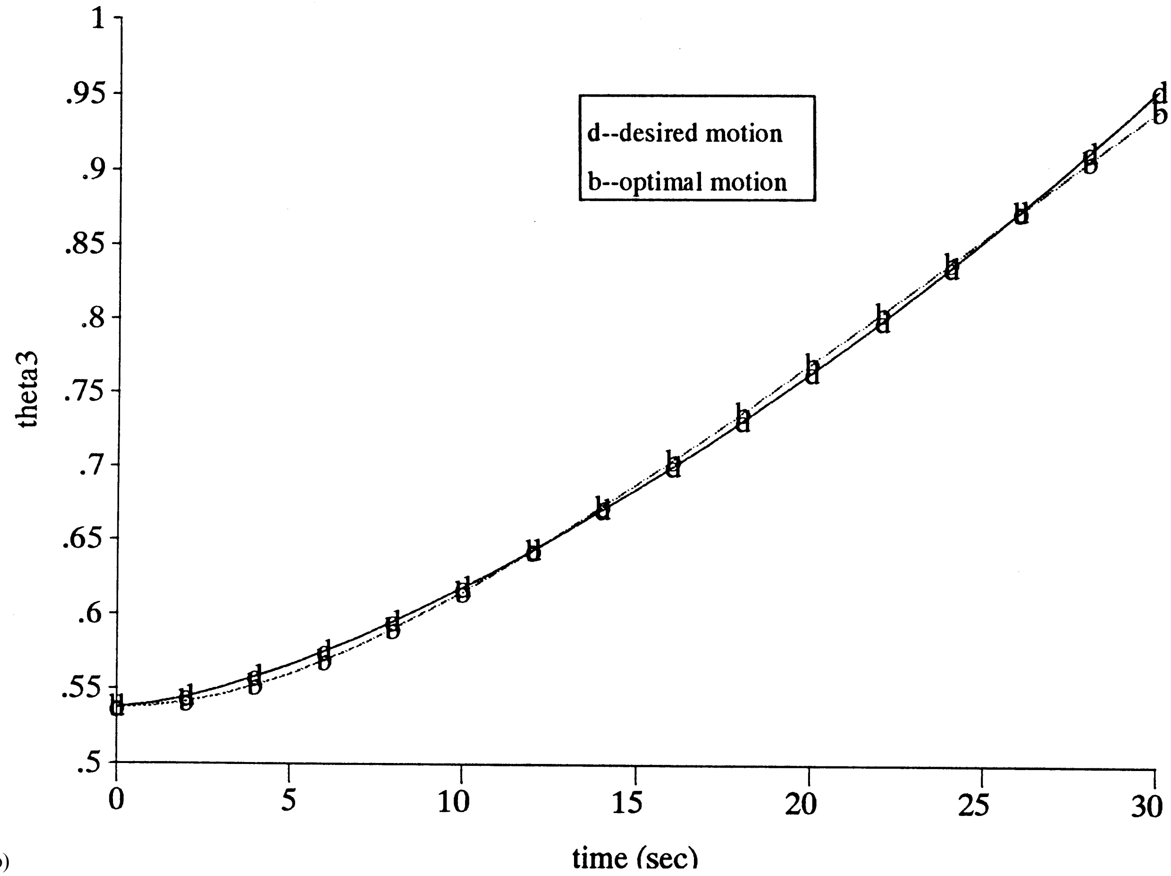
POSITION SYNTHESIS

A typical position synthesis problem may be stated as follows. Design a crank-rocker mechanism, as shown in Fig. 1, such that the output angular position, θ_3, of the rocker, follows a relation defined as a function of the input angular position of the crank, θ_1, as the crank rotates from θ_{10} to $(\theta_{10} + 90^\circ)$, i.e.,

$$\theta_{3i}^d = \theta_{30}^d + \frac{2}{3\pi}(\theta_{1i} - \theta_{10})^m \qquad (1)$$



(a)



(b)

Fig. 4. Comparison of the desired motion and optimal motion. (a) $m = 1$. (b) $m = 1.5$.

where the superscript 'd' represents the desired position. The exponent $m > 0$ can be varied to obtain different desired motions. θ_{10} and θ_{30}^d are right extreme angles for the crank and rocker respectively. They are defined in Fig. 1 (dashed line) as:

$$L_3^2 = (L_1 + L_2)^2 + L_4^2 - 2(L_1 + L_2)L_4 \cos(\theta_{10})$$
$$(L_1 + L_2)^2 = L_3^2 + L_4^2 - 2L_3L_4 \cos(180^\circ - \theta_{30}^d) \quad (2)$$

Performance measure

The performance measure is defined as the least squares error between the desired angular position and the actual angular position:

$$J(L_2, L_3) = \sqrt{\sum_{i=0}^{n} (\theta_{3i} - \theta_{3i}^d)^2 / (\theta_{3n}^d - \theta_{30}^d)} \quad (3)$$

By minimizing the performance measure $J(L_2, L_3)$ a best design, $J(L_2^*, L_3^*)$, can be achieved. In Fig. 2, values of the performance measure are plotted for L_3 in the range 5.5 to 9.3, with various constant values for L_2. The determination of the minimum performance measure J is, prima facie a two-dimension optimization problem, but, for simplification purposes, a best combination of L_2^* and L_3^* is found by a one-dimension search, in conjunction with the list-solving facility provided by TK.

TK Solver Program

TK has a well organized structure, subdividing the complete program into various task sheets. Figure 3(a) shows the TK Rule Sheet, with L_3 chosen as the variable to be optimized, and L_2 chosen as a list input variable. The search range for L_3 is determined by the Grashof conditions:

Lower limit:

$$\max(L_3 \geq L_1 + L_4 - L_2 \text{ or } L_3 \geq L_1 + L_2 - L_4)$$

Upper limit:

$$L_3 \leq L_2 + L_4 - L_1 \quad (4)$$

The search is accomplished by using the Golden search algorithm [4], as featured in the TK library function named Golden. Figure 3(b) shows a user-defined procedure function named POSPE called by Golden. The procedure function allows use of looping control and list variables, and within POSPE a rule function named POS, shown in Fig. 3(c), is invoked. Provided with arguments θ_1 and L_2, the function POS solves for θ_2 and θ_3, using the following rules derived from the geometry of Fig. 1:

$$r = \sqrt{L_1^2 + L_4^2 - 2L_1L_4 \cos(\theta_1)}$$
$$L_2^2 = r^2 + L_3^2 - 2rL_3 \cos(\alpha)$$
$$L_1^2 = r^2 + L_4^2 - 2rL_4 \cos(\beta)$$
$$L_2 \sin(\theta_2) = L_3 \sin(\theta_3) - L_1 \sin(\theta_1)$$
$$L_2 \sin(\theta_2) = L_4 + L_3 - L_1 \cos(\theta_1)$$
$$\theta_3 = \begin{cases} \pi - \alpha - \beta & \text{if} \quad 0 \leq \theta_1 \leq \pi \\ \pi - \alpha + \beta & \text{if} \quad \pi < \theta_1 \leq 2\pi \end{cases} \quad (5)$$

The Variable Sheet is shown in Fig. 3(d). It may be noted that the output variables *opt*L_3 and *pospe* are list variables; with given input variables L_1, L_2, L_4, n, and m, list solving can be conducted. Figure 3(e) shows a typical Table Sheet containing the solution for $m = 1.5$. From this sheet the best combination of L_2 and *opt*L_3 can be found. The TK Plot Sheet facilitates a graphical depiction of the results. Figure 4 shows two plots comparing the desired motion and the best output motion for $m = 1$ and $m = 1.5$, and it can be seen that a best fit is obtained for the latter case. The program is easily modified to accommodate L_2 as the variable to be optimized; L_3 then becoming a list variable.

VELOCITY SYNTHESIS

In a manner similar to that described above, a velocity synthesis can be conducted. A typical problem may be stated as follows: Given the crank angular velocity, ω_1, during the motion period τ defined by:

$$\omega_1 = \begin{cases} \omega_{1m} t/\tau & t \leq \tau/2 \\ 2\omega_{1m}(1 - t/\tau) & \tau/2 < t \leq \tau \end{cases} \quad (6)$$

where ω_{1m} is the maximum angular velocity of the crank, design a crank-rocker linkage such that the rocker angular velocity, ω_3, follows the pattern defined by:

$$\omega_3^d = 0.2\omega_1 \quad (7)$$

For this problem the performance measure can be defined as:

$$J(L_2, L_3) = \sqrt{\sum_{i=0}^{n} (\omega_{3i} - \omega_{3i}^d)^2} \quad (8)$$

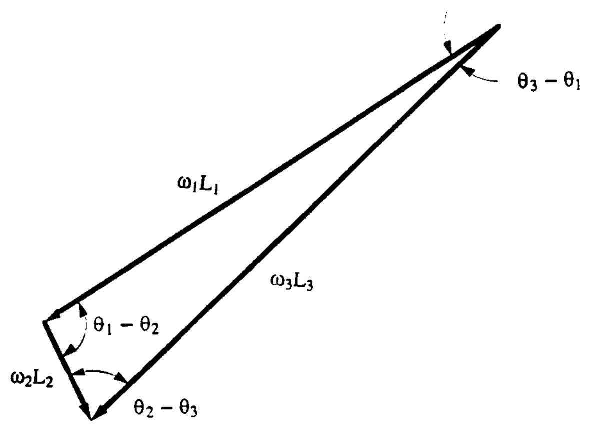


Fig. 5. Velocity vector diagram.

```
Rule
L21=L1 + L4 - L3          ; L2+L3 >= L1+L4   Grashof condition
L22=L1+L3-L4              ; L2+L4 >= L1+L3   Grashof condition
if (L21 > L22) then L2L=L21+.2 else L2L=L22+.2  ; lower limit of L2
L2U=(L3+L4-L1)-.2         ; L3+L4 >= L1+L2   Grashof condition, upper limt of L2
call Golden(L2L, L2U, 'VELPE; optL2, velpe)  ;Golden search to find optimal L2 for desired velocity
```

(a)

```
Comment:                 VELPE for evaluating performance measure
Parameter Variables:     L1,L3,L4,n,w1m,m
Input Variables:         L2
Output Variables:        velpe
Statement
theta10=acos(((L1+L2)^2+L4^2-L3^2)/(2*(L1+L2)*L4))
theta30=acos(((L1+L2)^2-L3^2-L4^2)/(2*L3*L4))
mn=0.5*n+1
velpe=0
for i=1 to n+1
't[i]=i-1
if (i <= mn) then 'w1[i]=2*w1m*(i-1)/n else 'w1[i]=w1m-2*w1m*(i-mn)/n
if (i <= mn) then 'theta1[i]=theta10+w1m*(i-1)^2/n else 'theta1[i]='theta1[mn]+w1m*((i-mn) - (i-mr
call POS('theta1[i], L2; 'theta2[i], 'theta3[i])
'w3d[i]=0.2*'w1[i]^m
call VEL(L2, 'w1[i], 'theta1[i], 'theta2[i], 'theta3[i]; 'w2[i], 'w3[i])
velpe=velpe+('w3[i]-'w3d[i])^2
next i
velpe=sqrt(velpe)
call statmsg("L2=  ", L2, "velpe=  ", velpe)
```

(b)

```
Comment:                 POS for evaluating position
Parameter Variables:     L1,L3,L4
Argument Variables:      theta1,L2
Result Variables:        theta2,theta3
Rule
r=sqrt(L1^2+L4^2-2*L1*L4*cos(theta1))
L2^2=r^2+L3^2-2*r*L3*cos(alpha)
L1^2=r^2+L4^2-2*L4*r*cos(beta)
if and (theta1 > 0, theta1 <= pi()) then theta3=pi() - alpha - beta else theta3=pi() - alpha + beta
theta2=atan2(L3*sin(theta3)-L1*sin(theta1), L4+L3*cos(theta3)-L1*cos(theta1))
```

(c)

Fig. 6. TK program for velocity optimization. (a) Rule Sheet. (b) Procedure Function VELPE. (c) Rule Function POS for velocity synthesis. (d) Rule Function VEL.

Comment:	VEL for evaluating velocity
Parameter Variables:	L1,L3,L4
Argument Variables:	L2,w1,theta1,theta2,theta3
Result Variables:	w2,w3

S Rule

(w2*L2)*sin(theta3-theta2)=(w1*L1)*sin(theta3-theta1)

(w3*L3)*sin(theta3-theta2)=(w1*L1)*sin(theta1-theta2)

(w2*L2)^2=(w1*L1)^2+(w3*L3)^2-2*(w1*L1)*(w3*L3)*cos(theta3-theta1)

(d)

Fig. 6 (Continued).

The velocity rules can be found from the vector diagram shown in Fig. 5:

$$\omega_3 L_3 \sin(\theta_3 - \theta_2) = \omega_1 L_1 \sin(\theta_1 - \theta_2)$$

$$\omega_2 L_2 \sin(\theta_2 - \theta_3) = \omega_1 L_1 \sin(\theta_3 - \theta_1)$$

$$(\omega_2 L_2)^2 = (\omega_1 L_1)^2 + (\omega_3 L_3)^2 - 2(\omega_1 L_1)(\omega_3 L_3)\cos(\theta_3 - \theta_1) \quad (9)$$

The Rule Sheet is shown in Fig. 6(a). For illustrative purposes L_2 is chosen for optimization, and L_3 is nominated as a list variable. A procedure function named VELPE, Fig. 6(b), first calls two rule functions POS and VEL, as shown in Fig. 6(c) and (d) respectively, and then computes the performance measure of equation (8). A comparison of the desired velocity, ω_3^d, and best velocity, ω_3^*, is shown in Fig. 7, with $L_3^* = 8$ and $L_2^* = 10.8235$. Optimization produces a curve closely fitting the desired one.

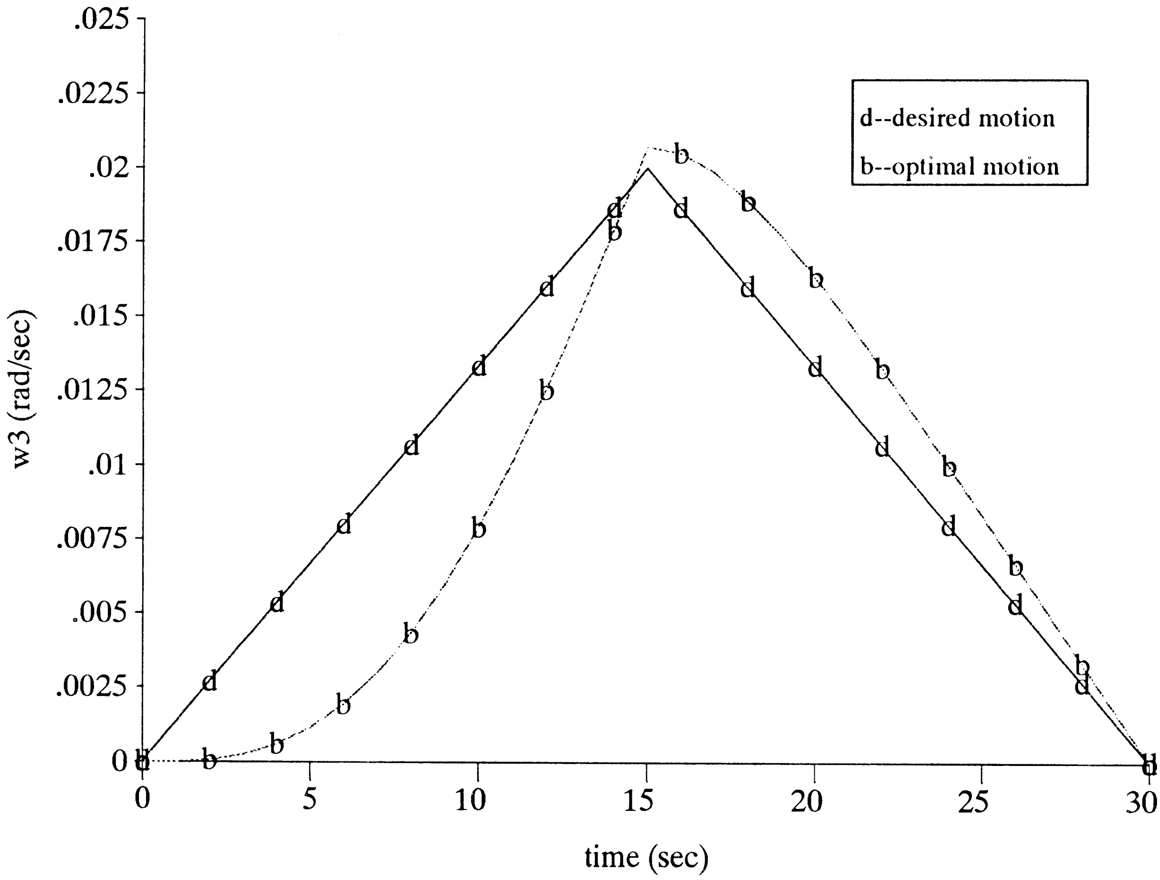


Fig. 7. Comparison of the desired velocity and optimal velocity for $L_2^* = 10.8235$ and $L_3^* = 8$.

CONCLUSIONS

Many unique features of TK Solver make it an ideal computational vehicle for mechanism analysis and synthesis. The work reported above illustrates a particular application. The relations between link positions and between link angular velocities are deduced from vector diagrams, and these expressions are used directly in TK rule functions. The difficulties associated with separating unknown from known, and of writing equations in sequential formulation, as required by conventional programming languages, are avoided. A two-dimension optimization is accomplished, using a one-dimension search algorithm combined with list solving.

The variety of utilities provided by TK, such as Plot Sheet, Table Sheet, and Variable Sheet, greatly facilitates problem solving and easy visualization. TK Solver makes this optimization problem straightforward and manageable, and the reported work can readily be modified to suit other design optimization situations.

REFERENCES

1. Robert L. Norton, *Design of Machinery*, McGraw-Hill, Inc., New York, NY (1993).
2. W. P. Boyle, Mechanism kinetics: pseudographics and TK! Solver, *Int. Jour. Appl. Engng. Ed.*, **4,** 427–434 (1993).
3. *TK Solver for Windows, User's Guide*, Universal Technical Systems, Inc., 1220 Rock Street, Rockford, IL 61101, USA (1994).
4. J. N. Sidal, *Optimal Engineering Design: Principles and Applications*, Marcel Dekker, New York, NY (1982).

W. Peter Boyle holds B.Sc. and Ph.D. degrees in Mechanical Engineering from The Queen's University of Belfast, is Professor of Engineering at Saint Mary's University, Halifax, and was previously a lecturer in the Department of Mechanical Engineering, University of Cape Town. He is the author of a text book *Introductory Fluid Mechanics*, and about twenty papers in a variety of topics in mechanical engineering. He has a current interest in the application of emergent software packages to engineering pedagogy, particularly in the area of mechanism analysis.

Kefu Liu holds a Bachelor's degree in Mechanical Engineering and a Master's degree in Applied Science from The Central South University of Technology, China. He was a lecturer at that university until commencing his doctoral work in 1988 at the Technical University of Nova Scotia, receiving his Ph.D. in Mechanical Engineering in 1992. For the past two years he has been an Assistant Professor at Saint Mary's University, Halifax, and at present is an Adjunct Assistant Professor in the Department of Mechanical Engineering at TUNS. His research interests are in the areas of vibrations, machine condition monitoring, robotics, dynamics of large space structures, having a number of publications in these topics.