RESEARCH ARTICLE OPEN ACCESS

Simulation of Script Writing Robot in RRL Configuration (SCARA)

S. Bhagavathi Shankar*, P.Raju*, M.Arun*
*(Assistant Professor Department of Mechanical Engineering, V.S.B.Engineering College, Karur - Tamilnadu)

ABSTRACT
In a typical education system dealing with kindergarten, it is a practice to hold the hands of the child to teach him to write. In the same scenario where there is a ratio of 1:40, teachers vs. students, it is impractical to give individual attention. A teaching assistance robot will be very useful in this scenario. This project is about giving a possible solution to this problem by implementing a script writing robot. A simulation of a RRL (Revolute-Revolute-Linear) Tamil-script writing robot is done using MATLAB10 software. The Tamil script is a very complex script. It involves geometric figures like circle, cycloid, spiral, ellipse, etc... Hence, by studying Tamil scripts, other language scripts can be easily adapted. By implementing these types of script writing robots, a substantial amount of resources in terms of money and time are reduced. It will help every child to have a firm grasp on their basic education and improves the overall quality of teaching and learning
Keywords: MATLAB,RRL, Spirals Tamil script.

I. PRESENT SYSTEM VERSUS PROPOSED SYSTEM

In the present system, there are number of teaching assistance programs that enable easy and efficient learning. But there are very few programs that are used as a teaching assistance for a Kindergarten. Especially in learning to write, the available assistance is meager. Hence the teaching is forced to happen in the traditional way of a teacher holding the hands of the child and teaching him to write.

Fig No:1.1 Present System

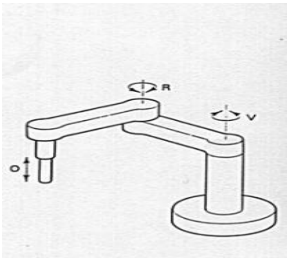


Fig No: 1.2 Proposed system

In the proposed system, a simple robot in the RRL (SCARA) configuration, which is typical in design of any SCARA assembly robot, is used as a teaching assistance in learning to write. This system when implemented can be a potential replacement of the **traditional** teaching methods.

II. INVERSE KINEMATICS

Given the angles at all of the robot's joints, what is the position of the hand? Imagine you are sitting at your table and you want to grasp a glass of water that is standing on the table. You will, of course, automatically modify the angles at your shoulder, elbow and wrist to position your hand around the glass. You will do so by taking into account the lengths of your arm and forearm, and the effect of different angles at your joints, until your visual senses inform you that your hand is close enough to the glass. This is an instance of an inverse kinematics problem: given a target position of the end-effectors, what are the (possibly many) values of the internal coordinates that satisfy it? One particularly useful instance of the inverse kinematics problem for proteins is the loop closure problem: given a flexible section of a protein (commonly referred to as a loop), are there any values for its internal coordinates that ensure the loop endpoints connect two other protein domains? Problems such as this usually require the definition and understanding of the protein's forward kinematics first, for which we develop a model next.

2.1 Inverse Kinematics – Link 3

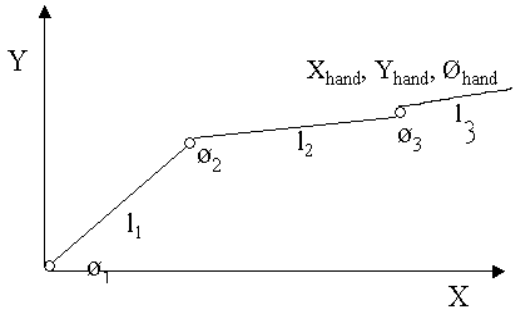


Fig 2.1- Inverse kinematics with link 3

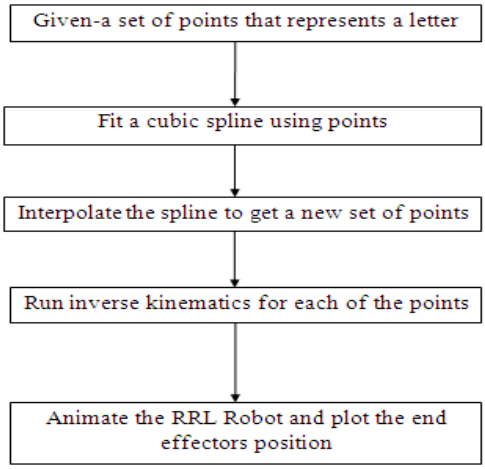
If the robot has two link of length l1, l2, and l3, & three angles with θ1, θ2, θ3

$x = l1 \cos \theta1 + l2 \cos (\theta1 + \theta2) + l3 \cos (\theta1 + \theta2 + \theta3)$ – (1)

$y = l1 \sin \theta1 + l2 \sin (\theta1 + \theta2) + l3 \sin (\theta1 + \theta2 + \theta3)$ – (2)

$ø = \theta1 + \theta2 + \theta3$

III. METHODOLOGY



IV. CONCEPT GENERATION AND SELECTION

The robot was called Selective Compliance Assembly Robot Arm, SCARA. Its arm was rigid in the Z-axis and pliable in the XY-axes, which allowed it to adapt to holes in the XY-axes. By virtue of the SCARA's parallel-axis joint layout, the arm is slightly compliant in the X-Y direction but rigid in the 'Z' direction, hence the term: Selective Compliant. This is advantageous for many types of assembly operations, i.e., inserting a round pin in a round hole without binding.

The second attribute of the SCARA is the jointed two-link arm layout similar to our human arms, hence the often-used term, Articulated. This feature allows the arm to extend into confined areas and then retract or "fold up" out of the way. This is advantageous for transferring parts from one cell to another or for loading/ unloading process stations that are enclosed. SCARA's are generally faster and cleaner than comparable Cartesian systems. Their single pedestal mount requires a small footprint and provides an easy, unhindered form of mounting. On the other hand, SCARA's can be more expensive than comparable Cartesian systems and the controlling software requires inverse kinematics for linear interpolated moves. This software typically comes with the SCARA though and is usually transparent to the end-user. SCARA is the chosen configuration. It gives high stability and convenience required for a script writing robot.

Fig no:4.1 SCARA Model

V. LANGUAGE AND METHODS

5.1 Tamil Script and Interpolation

The Tamil script has been chosen because of the complexity of the script. By simulating a robot for such complex curves, it can be easily adapted to other languages It is defined as a method of constructing a new data points within the range of discrete set of known data points. Interpolation is a method of constructing new data points within the range of a discrete set of known data points.

A different problem which is closely related to interpolation is the approximation of a complicated function by a simple function. Suppose we know the function but it is too complex to evaluate efficiently. Then we could pick a few known data points from the complicated function, and try to interpolate those data points to construct a simpler function. Of course, when using the simple function to calculate new data points we usually do not receive the same result as when using the original function, but depending on the problem domain and the interpolation method used the gain in simplicity might offset the error.

It should be mentioned that there is another very different kind of interpolation in mathematics, namely the "interpolation of operators. There are also many other subsequent results. Polynomial interpolation: It is a form of curve fitting where a polynomial is constructed that will go exactly through a given set of data points.

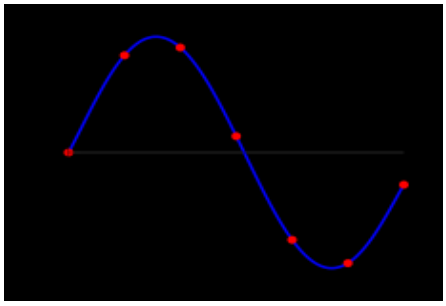


Fig no: 5.1 Spline interpolation

VI. MATLAB CODES

```
axis([0, 10, 0, 10 ])
[x,y]=ginput(30)
r2d=180/ pi
for n = 1:1:30
ee= [x(n) y(n)];
linklen= [6  4];
theta = InvKin(ee, linklen);
angles= theta * r2d
disp (angles );
end
yy= spline (x, y, xx)
function [theta] = InvKin(ee, linklen)
 % DESCRIPTION: Function to calculate the joint angles
% given the end effector position and link lengths.
%     theta -- The joint angles
%     linklin -- The link lengths

%     ee -- The end effector position
 D = (ee(1)^2 + ee(2)^2 - linklen(1)^2 - linklen(2)^2) / (2*linklen(1)*linklen(2));

% calculate theta2
theta(2) = acos(D);

% calculate theta1
theta(1) = atan (ee(2)/ee(1)) – atan (linklen(2)*sin(theta(2)) / (linklen(1) + linklen(2)*cos(theta(2))));
```

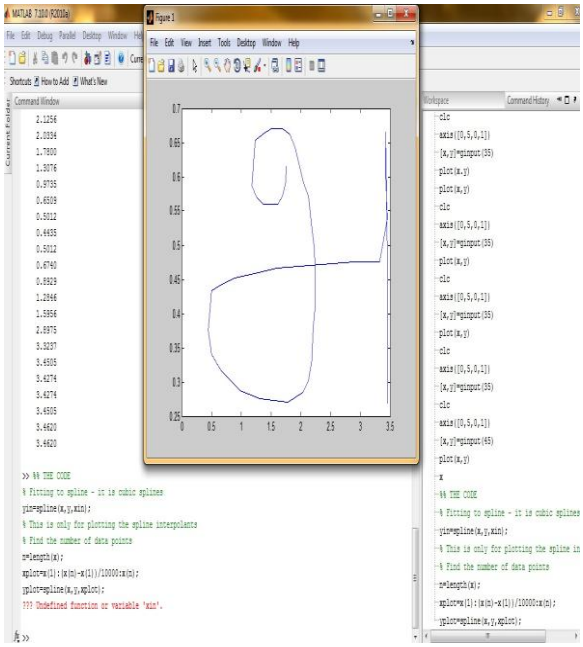


Fig no: 6.1 MATLAB Plotted area

VII. CONVERSION OF VALUES TO ANGLES

SL. NO	X-VALUES	Y-VALUES	ANGLE(θ)-DEGRE	
			θ1	θ2
1.	4.2972	6.9737	30.7431	71.6662
2.	4.5507	6.6228	26.7937	74.8182
3.	4.3664	6.0673	22.0259	85.3676
4.	3.7673	6.0088	23.6049	92.0323
5.	3.5138	6.2427	26.6867	90.8139
6.	3.3295	6.7982	32.2258	83.6587
7.	3.5599	7.2368	35.2975	74.2305
8.	3.9286	7.4415	36.2356	66.9289
9.	4.7120	7.0029	30.3339	66.3649
10.	5.3111	5.4825	14.6077	82.5007
11.	5.4724	4.4883	4.9749	92.2787
12.	5.4032	2.7924	-11.3296	108.2196
13.	4.2281	2.5292	-10.6211	125.2837
14.	2.2005	2.9971	13.0106	142.6858
15.	1.3479	4.2251	30.4979	132.3430
16.	1.0714	6.1842	42.2236	105.2278
17.	2.5691	6.0380	30.1586	100.7365
18.	4.0668	6.0380	22.7165	88.8103
19.	5.3111	5.8918	18.5583	76.8491
20.	6.6475	5.8918	19.6543	55.9121
21.	7.3387	5.8918	22.7791	40.3700
22.	7.4078	7.4123	30.4877	36.0485
23.	7.2465	8.2018	28.0966	50.3782
24.	7.3848	6.4181	31.2868	24.3574
25.	7.4309	5.2193	15.1794	50.6125
26.	7.4770	4.1374	4.0728	64.0247

27.	7.3387	3.2310	-5.0670	75.1573
28.	7.3618	3.6988	-0.5932	70.6848
29.	7.3618	4.2836	5.0790	64.6582
30.	7.2926	4.8684	10.7657	58.7740

VIII. CONCLUSION

Thus the given set of points that forms a letter has been converted into the joint angles of the robot. From the angles the simulation is done. This program can be used directly on a prototype. It is highly flexible and can be used for any script

IX. FUTURE RESEARCH

The splines in MATLAB can be fit only in either increasing or decreasing order. But since a letter goes in both the direction, we have to separate the points whenever it changes direction. With the subset of points, splines can be fit for each set. This above assumption requires further research.

REFERENCES

[1] http://www.learnaboutrobots.com/
[2] http://medialab.di.unipi.it/web/IUM/Waterloo/node157.html
[3] http://www.alglib.net/interpolation/spline3.php
[4] Brian R.Hunt, Ronald L. Lipsman and Jonathan M. Rosenberg. A Guide to MATLAB for beginners and experienced users. Newyork: Cambridge university press 2001
[5] Advances in Robot Kinematics - Jadran Lenarcic and Bernard Roth