مجله مهندسی مکانیک مدرس، آبان 1394، دوره 15، شماره 8، ص‌ص 153-163

ماهنامه علمی پژوهشی

مهندسی مکانیک مدرس

mme.modares.ac.ir

طراحی مسیر به کمک نقاط میانی و روش جرک- مینیمم در حضور موانع استاتیک برای بازوی رباتیکی 7 درجه آزادی

حسین رضایی‌فر[1]، فرشید نجفی[2*]

1- کارشناس ارشد، مهندسی مکانیک، دانشگاه تهران، تهران
2- استادیار، مهندسی مکانیک، دانشگاه تهران، تهران
* تهران، 4563- 11155، farshid_najafi@ut.ac.ir

اطلاعات مقاله

مقاله پژوهشی کامل
دریافت: 05 اردیبهشت 1394
پذیرش: 15 خرداد 1394
ارائه در سایت: 03 تیر 1394

کلید واژگان:
طراحی مسیر بهینه
ربات‌های سری
الگوریتم عدم برخورد
روش جرک- مینیمم

چکیده

به منظور استفاده از ربات‌ها برای انجام کارهای صنعتی، طراحی مسیر مناسب امری ضروری می‌باشد. اجرا مسیر توسط ربات هنگام حضور موانع، طراحی مسیر را دشوارتر می‌نماید. به علاوه طراحی مسیر برای هر عمل خاص امری زمانبر بوده و نیاز به افرادی متخصص داشته تا مسیر برای هر عمل صنعتی تعریف شود. این مقاله با استفاده از روش جرک- مینیمم، منحنی‌های بی‌اسپیلاین، نقاط میانی و الگوریتمی برای عدم برخورد با مانع به دنبال آن است تا به طور خودکار برای یک بازوی صنعتی 7 درجه آزادی که در محیط نرم‌افزاری سیممکانیک شبیه‌سازی گردیده، مسیری ایمن و مطلوب کاربر طراحی نماید. کاربر در ابتدا چند نقطه‌ی میانی حرکت ربات را با استفاده از سنسور کینکت مشخص نموده، سپس با استفاده از تلفیق روش جرک- مینیمم و منحنی‌های بی‌اسپیلاین و با احتساب محدودیت‌های سینماتیکی منحنی مسیر ترسیم گردیده، سپس این منحنی وارد الگوریتم عدم برخورد با مانع شده و مسیر نهایی بدون برخورد ایجاد می‌شود. الگوریتم عدم برخورد با استفاده از سینماتیک معکوس ربات به تصحیح مسیر می‌پردازد. از مزیت‌های روش پیشنهادی می‌توان به تسهیل طراحی مسیر برای کاربر و ایجاد مسیری هموار برای بازوی رباتیکی نام برد.

Path planning using via-points and Jerk-minimum method with static obstacles for a 7 DOF manipulator

Hossein Rezaeifar, Farshid Najafi*

Department of Mechanical Engineering, University of Tehran, Tehran, Iran
* P.O.B. 11155-4563 Tehran, Iran, farshid_najafi@ut.ac.ir

ARTICLE INFORMATION

Original Research Paper
Received 25 April 2015
Accepted 05 June 2015
Available Online 24 June 2015

Keywords:
Path planning optimization
Serial robots
Obstacle avoidance
Jerk-minimum method

ABSTRACT

In order to utilize robots for industrial tasks, designing a suitable path is necessary. Executing the path in the presence of obstacles makes the path planning task a difficult one. In addition, path planning is a time consuming task and needs expertise to define a certain path for each industrial job. In this paper, by use of Jerk-minimum method, B-Spline curves, via-point, and obstacle avoidance algorithm are used to automatically generate a suitable and safe path for a simulated 7 degrees of freedom industrial manipulator. A user determines via-points for robot trajectory using a Kinect sensor then employing a combination of Jerk-minimum method and B-Spline curves, a path is generated. This path is checked by an obstacle avoidance algorithm, and a final path is generated. The obstacle avoidance algorithm uses the inverse kinematic equation of the robot to modify the robot trajectory. One advantage of the proposed method is it facilitates trajectory planning for the user and creates a smooth trajectory for the robotic arm.

1- مقدمه

تعیین چگونگی حرکت ربات‌های سری در محیط کارگاهی اهمیت فراوانی دارد. طراحی مسیر این ربات‌ها را می‌توان به دو دسته‌ی کلی طراحی در محیط کارتزین[1] و طراحی در محیط مفصلی[2] تقسیم نمود. طراحی در محیط مفصلی به دلیل دو بعدی بودن آن و اعمال سریع‌تر توابع بر روی منحنی زوایه‌ی مفاصل ساده‌تر می‌باشد.

تاکنون محققان، توابع ریاضی متعددی مانند چندجمله‌ای‌ها، منحنی‌های بزیر[3]، منحنی‌های بی‌اسپیلاین[4] و منحنی‌های کیوبیک‌اسپیلاین[5] را در طراحی مسیر[6] استفاده کرده‌اند. پل و ژنگ [1] در سال 1984 اولین افرادی بودند که منحنی‌های چند جمله‌ای را برای طراحی مسیر در محیط مفصلی پیشنهاد کردند. بعد از آن‌ها کارهای متعددی توسط کیوو [2]، پارکین [3]،

1- Cartesian space
2- Joint space

3- Bezier
4- B-Spline
5- Cubic spline
6- Trajectory generation

برای ارجاع به این مقاله از عبارت ذیل استفاده نمایید:

Please cite this article using:

H. Rezaeifar, F. Najafi, Path planning using via-points and Jerk-minimum method with static obstacles for a 7 DOF manipulator, *Modares Mechanical Engineering,* Vol. 15, No. 8, pp. 153-163, 2015 (In Persian)

کریگ [4]، اسپانگ و همکارانش [5]، جزار [6] بر روی توابع چندجمله‌ای درجه‌ی سه انجام شد. اما در تمامی این موارد، جرک[1] دچار عدم پیوستگی بوده و محدودیت سینماتیکی آن‌ها نیز تضمین‌شده نمی‌باشد. نقاط میانی می‌توانند کمک بزرگی در راستای طراحی مسیر باشند. کریگ [4] پیشنهاد پیوند زدن 2 منحنی چندجمله‌ای درجه‌ی 3 برای ایجاد مسیر در فضای مفصلی با یک نقطه‌ی میانی را داد. بعدها انجلز [7] منحنی درجه 7 را برای طراحی با دو نقطه‌ی میانی ارائه نمود. همان‌طور که مشخص است، تمامی روش‌های استفاده‌کننده از منحنی‌های چندجمله‌ای در محدود کردن مشتقات مسیر مناسب نبوده‌اند و همچنین حجم محاسباتی بالایی برای بدست آوردن ضرایب چندجمله‌ای در آن‌ها نیاز است.

در استفاده از منحنی‌های بزیر، ونگ و هورنگ [8] با استفاده از بی‌اسپیلاین و ایجاد محدودیت در سرعت و شتاب و جرک به طراحی مسیر در محیط کارتزین دو بعدی پرداختند. اما آنچه که بیشتر از موارد یاد شده مورد استفاده قرار می‌گیرد، ترکیب یکی از این روش‌ها با روش‌های بهینه‌سازی است. به دلیل آنکه در صنعت مسئله‌ی زمان از اهمیت بالایی برخوردارنبوده و بیشتر کارهای انجام شده در راستای مینیمم کردن زمان است. کنستانتینسکو و کرافت [9] با استفاده از تابع هدف رابطه‌ی (1) برای زمان مینیمم و با کمک از دینامیک بازوهای صنعتی و محدودیت گشتاور موتور و مشتق آن به طراحی مسیر ربات‌های سری پرداختند.

$$\min J = \int_0^{t_f} 1 \, dt \qquad (1)$$

که در آن:

J- مجموع جرک در طول مسیر

t_f- کل زمان حرکت ربات

می‌باشند. این روش‌ها با محاسبات زیادی همراه بوده و اگر در آن‌ها چندین نقطه میانی موجود باشد تمامی سرعت‌های میانی بایستی معلوم باشند. بنابراین این الگوریتم‌ها بصورت خودکار نمی‌تواند مسیر گذرنده از نقاط دلخواه را به دست دهند. گاسپارتو و زانوتو [10] با تلفیقی از روش جرک- مینیمم و زمان‌مینیمم با کمک توابع بی‌اسپیلاین به طراحی مسیر پرداختند. در این کار زمان توسط الگوریتم تعیین شده‌است. روش جرک- مینیمم با ایجاد مسیری هموار به گونه‌ای که به عملگرهای ربات تنش بیش از حد وارد نشود، روشی مناسب برای طراحی مسیر در زمان مشخص است. هیوانگ و همکارانش [11] با استفاده از الگوریتم ژنتیک و محدودیت‌های سینماتیکی و دینامیکی بازوی مکانیکی به ایجاد مسیری هموار پرداختند که مینیمم نمودن بیشترین جرک در طول مسیر معیار اصلی در این طراحی بوده‌است. الگوریتم ژنتیک بسیار زمانبر و طولانی بوده و همچنین نقاط میانی مسیر در این کار توسط روش بیان‌شده در [11] تعیین می‌شوند و کاربر نقشی در تعیین آن‌ها ندارد. بهتر آن است که کاربر بتواند نقاط میانی مسیر حرکت را تعیین نموده تا حرکتی مطلوب کاربر نیز انجام شود.

برای طراحی مسیر ایمن بازوی رباتیکی، باید به این نکته توجه داشت که لینک‌های[2] ربات در طول مسیر برخوردی با موانع اطراف نیز نداشته‌باشند. بنابراین این مسئله می‌تواند با مسائل بیان‌شده تلفیق گشته و مسئله جدیدی در طراحی مسیر مطرح نماید. در سال 1994 گلاسیوس و همکارانش [12] به طراحی مسیر ربات دو درجه آزادی صفحه‌ای به کمک شبکه‌ی عصبی هاپفیلد[3] پرداختند. در این روش ربات از نقطه‌ی ابتدایی به نقطه‌ی هدف به گونه‌ای هدایت شد تا در طول مسیر به موانع موجود در فضای موقعیتی[4] ربات که بصورت موضعی دایره‌ای تعریف می‌شدند، برخوردی نداشته‌باشد. در این کار موانع به گونه‌ای تعریف شدند تا دسترسی به آن‌ها غیرممکن باشد. در سال 2000 یانگ و منگ [13] به وسیله‌ی شبکه‌ی عصبی به طراحی مسیری بدون برخورد با مانع دینامیک در فضای دو بعدی پرداختند. شبکه‌ی عصبی در این کار نقش شناساگر مانع را ایفا نمود. کلیه‌ی مطالعات فوق نشان می‌دهد که این روش‌ها برای ربات‌های صفحه‌ای[5] انجام گرفته و در هیچکدام ابعاد لینک‌ها در نظر گرفته نشده‌است. پارک و همکارانش [14] با استفاده از روش PRM[6] و یادگیری مقاومتی[7] به آموزش ربات به منظور گذر از مانع در محیط‌های استاتیک و دینامیک پرداختند، روشی که در آن تنها مسیر گذر مجری نهایی ربات بررسی گردید و همچنین نقاط میانی مسیر در آن‌ها لحاظ نگردیده که می‌تواند از جامعیت این روش برای ربات‌های سری بکاهد.

روشی مرسوم برای ایجاد مسیر در فضای مفصلی ربات استفاده از ماتریس ژاکوبین[8] می‌باشد. در این روش، مسیر در فضای کارتزین از ابتدا توسط کاربر مشخص است. در ادامه، فرایند ایجاد مسیر کمک نموده تا سینماتیک معکوس ربات به گونه‌ای تحلیل شود تا در طول مسیر حرکت، ربات با موانع موجود در اطراف مسیر برخوردی نداشته باشد، در ضمن بهینه بودن مسیر نیز مورد توجه قرار می‌گیرد. در این راه محققین مختلف کارهای متعددی انجام داده و این مسئله‌ی بهینه‌سازی را به طرق مختلف حل نموده‌اند [15- 20].

در این مقاله، طراحی مسیر با استفاده از نقاط میانی که توسط کاربر و از طریق سنسور کینکت به ربات نشان داده شده، تلفیق روش جرک- مینیمم و توابع بی‌اسپیلاین و با احتساب محدودیت‌های سینماتیکی انجام می‌پذیرد. اهمیت استفاده از روش جرک- مینیمم علاوه بر آنکه زمان انجام عملیات را به کاربر می‌سپارد، سبب می‌شود تا تغییرات گشتاور در مفاصل محدود شود و این خود تنش وارد بر عملگرهای بازو را کاهش داده و به طور کل به ایمنی بازوی رباتیکی کمک می‌کند، همچنین به کمک این روش حرکتی هموار و هماهنگ برای بازوی رباتیکی حاصل می‌شود. پس از مراحل ذکرشده منحنی وارد الگوریتم عدم برخورد با مانع شده و مسیر هموار نهایی بدون برخورد ایجاد می‌شود. الگوریتم عدم‌برخورد با استفاده از سینماتیک معکوس ربات و بصورت بازگشتی، مسیری اصلاح‌شده بدست می‌دهد. بدین ترتیب روش ارائه‌شده در این مقاله در مقایسه با روش‌های یاد شده در این بخش و به طور همزمان مسیری مطلوب کاربر به همراه سرعت و شتاب و جرک محدود و بدون برخورد با موانع اطراف ایجاد می‌نماید. در این مقاله در ابتدا به توضیح سینماتیک مستقیم و معکوس ربات پرداخته و سپس نحوه‌ی نشان دادن نقاط میانی توضیح داده می‌شود. در قسمت بعد با استفاده از نقاط میانی و تلفیق روش جرک- مینیمم و توابع بی‌اسپیلاین به طراحی مسیری هموار پرداخته شده، به طوری که نقاط میانی را کاربر انتخاب می‌نماید. سپس به توضیح الگوریتم عدم برخورد با مانع جهت تصحیح مسیر طراحی‌شده پرداخته می‌شود. در انتهای مقاله نتایج شبیه‌سازی ربات در محیط سیممکانیک[9] از نرم‌افزار متلب[10] ارائه می‌شود تا ابعاد مختلف آن بیشتر مورد بحث قرار گیرد.

1- Jerk
2- Link
3- Hopfiled

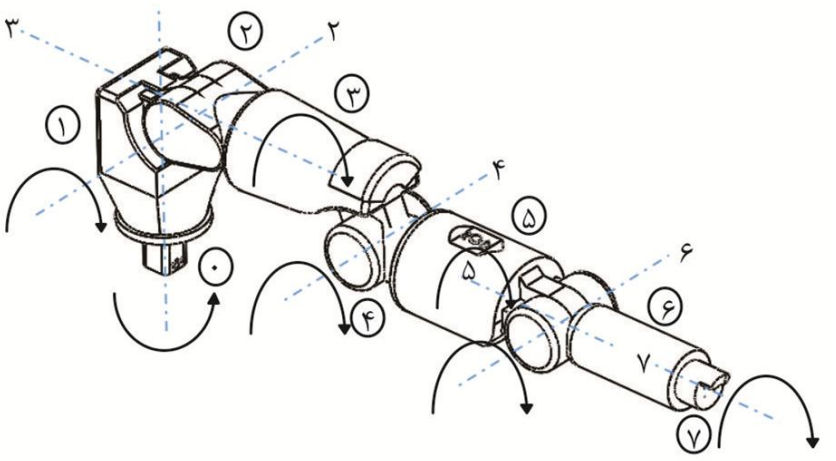
4- Configuration space
5- Planar robot
6- Probabilistic Roadmap
7- Reinforcement learning
8- Jacobian matrix
9- Simmechanics
10- Matlab

2- تحلیل سینماتیکی بازوی رباتیکی

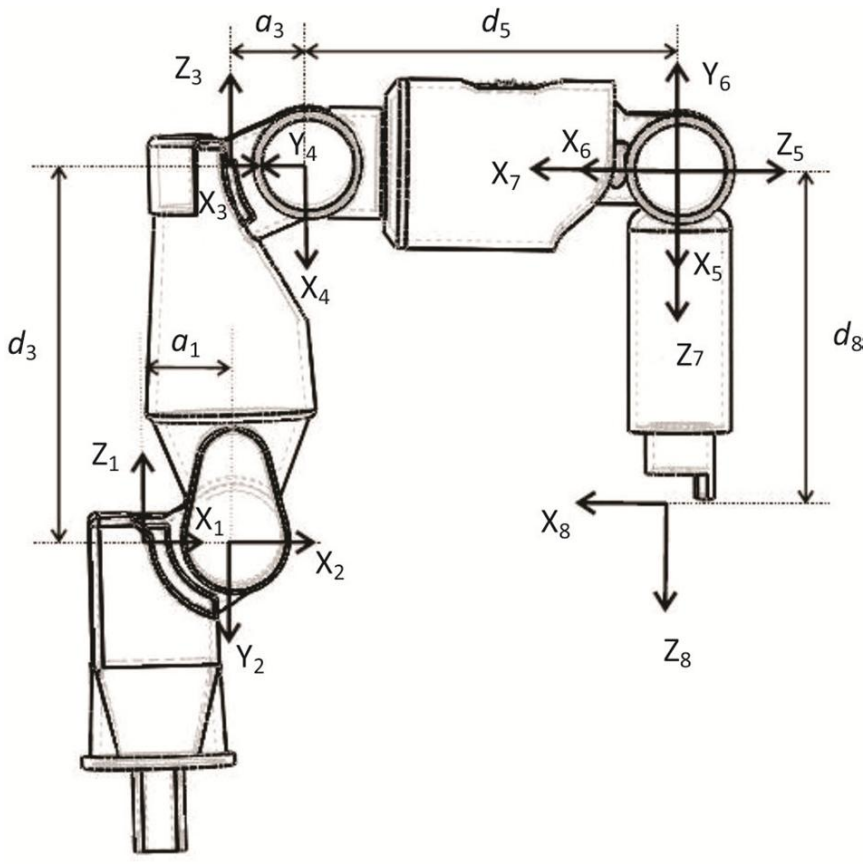
2-1- سینماتیک مستقیم

مدل بازوی طراحی‌شده در نرم‌افزار سالیدورکس[1] و نحوه‌ی حرکت هر مفصل در شکل 1 نشان داده شده‌است. در شکل 2، نمایش دوبعدی این بازوی رباتیکی و ابعاد اساسی برای تعیین پارامترهای دناویت- هارتنبرگ[2] نمایش داده شده‌اند. در جدول 1 مقادیر ابعاد موجود در شکل 2 بیان گردیده، با توجه به این ابعاد و دستگاه‌های متصل‌شده در شکل 2، پارامترهای دناویت- هارتنبرگ را مطابق [4] می‌توان در قالب جدول 2 نمایش داد.

پس از مشخص شدن جدول پارامترهای دناویت- هارتنبرگ، در مرحله‌ی



شکل 1 ربات طراحی‌شده در نرم‌افزار سالیدورکس (شماره هر لینک با عدد درون دایره نمایش داده شده‌است)



شکل 2 شکل دو بعدی بازوی رباتیکی به همراه ابعاد اصلی a_1، a_3، d_3، d_5، d_8

جدول 1 ابعاد اساسی بازوی رباتیکی

ابعاد	اندازه (متر)
a_1	0/1
a_3	0/1
d_3	0/45
d_5	0/45
d_8	0/4

جدول 2 پارامترهای دناویت- هارتنبرگ بازوی رباتیکی

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	-90	a_1	0	θ_2
3	90	0	d_3	θ_3
4	-90	a_3	0	θ_4
5	90	0	d_5	θ_5
6	-90	0	0	θ_6
7	90	0	0	θ_7
8	0	0	d_8	0

مشخص شود. در مورد ربات‌های سری، ماتریس انتقال همگن بین دستگاه مختصات i و دستگاه مختصات $i-1$ بصورت رابطه (2) با استفاده از پارامترهای دناویت- هارتنبرگ بدست می‌آید.

$$
{}_{i}^{i-1}T = \begin{bmatrix} \mathrm{C}\,\theta_i & -\mathrm{S}\,\theta_i & 0 & a_{i-1} \\ \mathrm{S}\,\theta_i\,\mathrm{C}\,\alpha_{i-1} & \mathrm{C}\,\theta_i\,\mathrm{C}\,\alpha_{i-1} & -\mathrm{S}\,\alpha_{i-1} & -\mathrm{S}\,\alpha_{i-1}\,d_i \\ \mathrm{S}\,\theta_i\,\mathrm{S}\,\alpha_{i-1} & \mathrm{C}\,\theta_i\,\mathrm{S}\,\alpha_{i-1} & \mathrm{C}\,\alpha_{i-1} & \mathrm{C}\,\alpha_{i-1}\,d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)
$$

ماتریس انتقال دستگاه 8 نسبت به دستگاه صفر به صورت رابطه (3) تعریف می‌شود.

$$
{}_{8}^{0}T = {}_{1}^{0}T\,{}_{2}^{1}T\,{}_{3}^{2}T\,{}_{4}^{3}T\,{}_{5}^{4}T\,{}_{6}^{5}T\,{}_{7}^{6}T\,{}_{8}^{7}T \quad (3)
$$

با جایگذاری ماتریس‌های انتقال همگن در رابطه‌ی (3) و ضرب ماتریس- ها، ماتریس انتقال بین دستگاه 0 و دستگاه 8، به صورت (4) بدست می‌آید.

$$
{}_{8}^{0}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)
$$

که در آن قسمت $\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$ موقعیت دستگاه 8 نسبت به دستگاه صفر را نشان می‌دهد و قسمت $\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$ دوران دستگاه 8 نسبت به دستگاه صفر را نشان می‌دهد.

2-2- سینماتیک معکوس

بازوی رباتیکی مورد نظر در این مقاله دارای هفت درجه آزادی می‌باشد، در حالی که برای رسیدن به یک موقعیت و جهت مشخص در فضا به شش درجه آزادی نیاز است. برای تحلیل سینماتیک معکوس ربات و یافتن زوایای مفاصل ربات به هفت معادله‌ی مستقل نیاز بوده، ولی معادلات حاصل از سینماتیک مستقیم شش معادله‌ی مستقل در اختیار می‌گذارند. بنابراین به ازای یک موقعیت و جهت مشخص مجری نهایی ربات در فضا، تعداد بی‌نهایت جواب برای معادلات سینماتیک معکوس آن وجود دارد. بنابراین نیاز است که یک معادله مستقل دیگر بصورت تابعی از موقعیت مفاصل و یا سرعت‌های مفاصل، برای یافتن جواب‌های محدود در اختیار باشد. ساده‌ترین راه برای بدست آوردن معادله‌ی مستقل مورد نیاز این است که یکی از زوایای مفاصل را معلوم فرض شود. در اینجا فرض می‌شود که زاویه θ_1 معلوم باشد. با این فرض و با استفاده از حل پایپر[3] در سینماتیک معکوس بازوهای رباتیکی [21]، رابطه‌ی (3) را از سمت چپ در معکوس ماتریس ${}_{1}^{0}T$ و از سمت راست در معکوس ماتریس ${}_{8}^{7}T$ ضرب کرده تا نتیجه‌ی (5) حاصل شود.

$$
{}_{2}^{1}T(\theta_2)\,{}_{3}^{2}T(\theta_3)\,{}_{4}^{3}T(\theta_4)\,{}_{5}^{4}T(\theta_5)\,{}_{6}^{5}T(\theta_6)\,{}_{7}^{6}T(\theta_7) = {}_{1}^{0}T^{-1}(\theta_1)\,{}_{8}^{0}T\,{}_{8}^{7}T^{-1} \quad (5)
$$

1- Solidworks
2- Denavit-Hatenberg
3- Piper

ماتریس T^* بصورت رابطه (6) تعریف می‌شود.

$$T^* = {}^0_1T^{-1}\ {}^0_8T\ {}^7_8T^{-1} \quad (6)$$

با جایگذاری ماتریس‌های ${}^0_1T$، ${}^7_8T$ و ${}^0_8T$، ماتریس T^* به صورت رابطه (7) قابل دسترس خواهد بود.

$$T^* = \begin{bmatrix} {r_{11}}^* & {r_{12}}^* & {r_{13}}^* & x \\ {r_{21}}^* & {r_{22}}^* & {r_{23}}^* & y \\ {r_{31}}^* & {r_{32}}^* & {r_{33}}^* & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

همان‌طور که واضح است، با فرض معلوم بودن زاویه‌ی θ_1، ماتریس T^* به طور کامل مشخص می‌باشد. بنابراین با حل معادله‌ی (5) می‌توان زوایای θ_2 تا θ_7 را بصورت تابعی از زاویه‌ی θ_1 و موقعیت دستگاه مختصات (8) بدست آورد. محورهای Z_5، Z_6 و Z_7 متقاطع می‌باشند و مبدا این سه دستگاه در یک نقطه قرار دارد. بنابراین می‌توان نوشت.

$${}^1P_5 = {}^1P_7 = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = {}^1_2T\ {}^2_3T\ {}^3_4T\,{}^4P_5 \quad (8)$$

با استفاده از رابطه‌ی (8) و سینماتیک مستقیم ربات می‌توان زوایای θ_2، θ_3 و θ_4 را محاسبه نمود. ابتدا زاویه‌ی θ_4 بدست می‌آید.

$$\theta_4 = \text{Atan2}(B, A) \pm \text{Atan2}(\sqrt{A^2 + B^2 - C^2}, C) \quad (9)$$

که در آن

$$A = 2d_3d_5,\ B = 2a_3d_5$$
$$C = (x - a_1)^2 + y^2 + z^2 - {a_3}^2 - {d_3}^2 - {d_5}^2 \quad (10)$$

برای محاسبه زاویه‌ی θ_3

$$\text{S}\,\theta_3 = \frac{y}{a_3 + d_5\,\text{S}\,\theta_4} \rightarrow \text{C}\,\theta_3 = \pm\sqrt{1 - \text{S}\,{\theta_3}^2} \quad (11)$$

$$\theta_3 = \text{Atan2}(\text{S}\,\theta_3, \text{C}\,\theta_3) \quad (12)$$

با معلوم بودن زوایای θ_3 و θ_4، زاویه‌ی θ_2 بصورت رابطه‌ی (13) بدست می‌آید.

$$\theta_2 = \text{Atan2}(\text{S}\,\theta_2, \text{C}\,\theta_2) \quad (13)$$

که در آن

$$\text{C}\,\theta_2 = \frac{(a_3 + d_5\,\text{S}\,\theta_4)\,\text{C}\,\theta_3\,(x - a_1) + (d_5\,\text{C}\,\theta_4 + d_3)z}{\left[\left((a_3 + d_5\,\text{S}\,\theta_4)\,\text{C}\,\theta_3\right)^2 + (d_5\,\text{C}\,\theta_4 + d_3)^2\right]}$$
$$\text{S}\,\theta_2 = \frac{(d_5\,\text{C}\,\theta_4 + d_3)(x - a_1) - (a_3 + d_5\,\text{S}\,\theta_4)\,\text{C}\,\theta_3\,z}{\left[\left((a_3 + d_5\,\text{S}\,\theta_4)\,\text{C}\,\theta_3\right)^2 + (d_5\,\text{C}\,\theta_4 + d_3)^2\right]} \quad (14)$$

در مرحله بعد، زوایای θ_5، θ_6 و θ_7 بدست می‌آیند. در این مرحله تنها لازم است که از بخش دورانی ماتریس انتقال ${}^0_8T$ استفاده شود. بدین منظور با مراجعه به رابطه‌ی (4)، بخش دورانی این رابطه بصورت رابطه‌ی (15) در نظر گرفته می‌شود.

$${}^0_1R\ {}^1_2R\ {}^2_3R\ {}^3_4R\ {}^4_5R\ {}^5_6R\ {}^6_7R\ {}^7_8R = {}^0_8R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (15)$$

با ضرب رابطه‌ی (15) از سمت راست در معکوس ماتریس ${}^7_8R$ و از سمت چپ در معکوس ماتریس‌های ${}^0_1R$، ${}^1_2R$، ${}^2_3R$ و ${}^3_4R$ نتیجه‌ی (16) حاصل می‌شود.

$${}^4_5R\,(\theta_5)\ {}^5_6R(\theta_6)\ {}^6_7R(\theta_7) = {}^3_4R^{-1}\ {}^2_3R^{-1}\ {}^1_2R^{-1}\ {}^0_1R^{-1}\ {}^0_8R\ {}^7_8R^{-1} \quad (16)$$

سمت چپ رابطه‌ی (16) به طور کامل معلوم بوده و ماتریس R^{**} به صورت رابطه‌ی (17) تعریف می‌شود.

$$R^{**} = {}^3_4R^{-1}\,{}^2_3R^{-1}\,{}^1_2R^{-1}\,{}^0_1R^{-1}\,{}^0_8R\,{}^7_8R^{-1} = \begin{bmatrix} {r_{11}}^{**} & {r_{12}}^{**} & {r_{13}}^{**} \\ {r_{21}}^{**} & {r_{22}}^{**} & {r_{23}}^{**} \\ {r_{31}}^{**} & {r_{32}}^{**} & {r_{33}}^{**} \end{bmatrix} \quad (17)$$

رابطه‌ی (17) با سینماتیک مستقیم ربات قابل محاسبه است. با استفاده از روابط (16) و (17) زوایای θ_5، θ_6 و θ_7 قابل محاسبه می‌باشند. زاویه θ_6 با رابطه‌ی (18) بدست می‌آید.

$$\theta_6 = \text{Atan2}(\text{S}\,\theta_6, \text{C}\,\theta_6) \quad (18)$$

که در آن

$$\text{S}\,\theta_6 = \pm\sqrt{{r_{21}}^{**2} + {r_{22}}^{**2}} \quad (19)$$

$$\text{C}\,\theta_6 = -{r_{23}}^{**} \quad (20)$$

پس از بدست آمدن زاویه‌ی θ_6، در نظر گرفتن مولفه‌های ${r_{21}}^{**}$ و ${r_{22}}^{**}$ می‌توان زاویه‌ی θ_7 را نیز بدست آورد.

$$\theta_7 = \text{Atan2}(\text{S}\,\theta_7, \text{C}\,\theta_7) \quad (21)$$

که در آن

$$\text{S}\,\theta_7 = \frac{-{r_{22}}^{**}}{\text{S}\,\theta_6} \quad (22)$$

$$\text{C}\,\theta_7 = \frac{{r_{21}}^{**}}{\text{S}\,\theta_6} \quad (23)$$

در نهایت برای زاویه‌ی θ_5 می‌توان نوشت

$$\theta_5 = \text{Atan2}(\text{C}\,\theta_5, \text{C}\,\theta_5) \quad (24)$$

که در آن

$$\text{S}\,\theta_5 = \frac{{r_{33}}^{**}}{\text{S}\,\theta_6} \quad (25)$$

$$\text{C}\,\theta_5 = \frac{{r_{13}}^{**}}{\text{S}\,\theta_6} \quad (26)$$

همان‌طور که ملاحظه گردید برای زوایای θ_3، θ_4 و θ_6 دو جواب بدست آورده‌شد. بنابراین می‌توان گفت که با فرض یک θ_1 معلوم، تعداد $2^3 = 8$ جواب متمایز برای سینماتیک معکوس ربات یافت می‌شود. حال حالتی از سینماتیک معکوس بازوی رباتیکی با فرض معلوم بودن زاویه‌ی θ_1 و جهت-گیری مجری نهایی ربات به سمت پایین مورد بررسی قرار می‌گیرد. از آنجایی که این بازو برای عمل گرفتن و جابه‌جایی اجسام در خطوط برنامه‌ریزی می‌شود این فرض، فرضی معقول بوده و در ربات‌های مشابه، مانند ربات انسان‌نمای بکستر مورد استفاده قرار گرفته‌است. اگر بازوی ربات به سمت پایین باشد، آنگاه ماتریس انتقال ${}^0_8T$ در رابطه‌ی (4) بصورت رابطه‌ی (27) بدست می‌آید.

$${}^0_8T = \begin{bmatrix} \text{C}\,\theta & \text{S}\,\theta & 0 & p_x \\ \text{S}\,\theta & -\text{C}\,\theta & 0 & p_y \\ 0 & 0 & -1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (27)$$

همان‌طور که در رابطه‌ی (27) ملاحظه می‌گردد، برای مشخص شدن موقعیت دستگاه 8 نسبت به دستگاه صفر در این حالت، تنها به چهار متغیر p_x، p_y، p_z و θ نیاز می‌باشد. θ، زاویه‌ی بین محور X_0 و X_8 می‌باشد. با جایگذاری عناصر ماتریس ${}^0_8T$ رابطه‌ی (27) در روابط (5) تا (26) زوایای ربات بدست خواهد آمد.

در این مقاله از حل سینماتیک معکوس در الگوریتم گذر از مانع استفاده خواهدشد. زاویه θ_1 معلوم برای فرایند سینماتیک معکوس برابر با $\text{Atan2}(p_y, p_x) - \alpha$ در نظر گرفته می‌شود، به نحوی که $-40 < \alpha < 40$ باشد تا ضمن اینکه سمت و سویی که بازو نسبت به مجری نهایی ربات می‌گیرد نزدیک به حالت مستقیم بوده، تغییر زوایای دیگر نیز محدود نگردد.

3- طراحی مسیر

3-1- تعیین نقاط میانی مسیر توسط کاربر

در این مقاله از سنسور کینکت به منظور تعیین نقاط میانی مسیر حرکت

جهت ورود به فرایند طراحی مسیر استفاده می‌شود. روش کار بدین صورت است که کاربر به کمک دست راست خود برای نشان دادن نقاط دلخواه استفاده می‌کند و با حرکت دست چپ خود فرمان ثبت نقاط را می‌دهد. برای ثبت نقاط ابتدایی و انتهایی حرکت دست چپ مطابق شکل 3(الف) می‌باشد و سایر نقاط مطابق شکل 3(ب) ثبت می‌گردند. موقعیت نقاط نسبت به مرکز بدن انسان سنجیده می‌شوند. بعد از این مرحله، نقاط میانی وارد فرایند طراحی مسیر می‌گردند.

3-2- مسیر بهینه بر پایه‌ی جرک- مینیمم

برای طراحی یک مسیر مناسب با استفاده از روش جرک- مینیمم توابع هدف گوناگونی مطرح می‌شوند. در این مقاله از تابع هدف ارائه‌شده در [22] مطابق رابطه‌ی (28) استفاده شده‌است.

$$L = \min \sum_{i=1}^{N} \int_0^{t_f} (\dddot{q}_i)^2 \, dt \tag{28}$$

که در آن N تعداد مفاصل ربات سری، t_f زمان کل حرکت و $\dddot{q}_i$ جرک مفصل iام می‌باشند. باید این نکته را متذکر شد که همراه با این تابع هدف برای دستیابی به اهداف امنیت ربات و کاربر و محدودیت‌های سینماتیکی، قیدهایی نیز لحاظ می‌شود. به گونه‌ای که

$$\begin{aligned} &|\dot{q}_i(t)| \le VC_i , \quad i = 1, \dots, N \\ &|\ddot{q}_i(t)| \le AC_i , \quad i = 1, \dots, N \\ &|\dddot{q}_i(t)| \le JC_i , \quad i = 1, \dots, N \end{aligned} \tag{29}$$

که در آن $\dot{q}_i$، $\ddot{q}_i$ و $\dddot{q}_i$ به ترتیب سرعت، شتاب و جرک مفصل iام و VC_i، AC_i و JC_i به ترتیب حدود سرعت، شتاب و جرک مفصل iام می‌باشند. همان‌طور که مشاهده می‌شود عمل بهینه‌سازی جرک در محدوده‌ی سینماتیکی مورد نظر کاربر صورت می‌گیرد. هدف نهایی از عمل بهینه‌سازی، برداری است که به کمک آن مدت زمانی را که ربات باید برای هر قطعه منحنی تعیین‌شده توسط کاربر صرف نماید، مشخص نموده و شرایط ایجاد منحنی موقعیت زاویه‌ای مفاصل را فراهم آورد. انتگرال‌گیری موجود در بهینه‌سازی زمانی انجام می‌شود که مسیری تعیین شده‌باشد. انتخاب مناسبی برای ایجاد مسیر می‌باشند.

3-3- ترکیب تابع مسیر بی‌اسپیلاین و بهینه‌سازی جرک- مینیمم

در این مقاله برای طراحی مسیر از توابع بی‌اسپیلاین درجه 5 استفاده می‌شود تا ضمن بدست آمدن منحنی هموار برای زوایای مفاصل، منحنی سرعت، شتاب نیز پیوسته باشند. عدم پیوستگی سرعت، شتاب صدمات سنگینی به سیستم رباتیکی وارد می‌آورد. تابع بی‌اسپیلاین درجه d به صورت ترکیب خطی از توابع چند جمله‌ای $N_{i,d}(t)$ تعریف می‌شود که این توابع چندجمله‌ای به وسیله‌ی نقاط کنترلی c_i وزن‌گذاری می‌شوند. این تابع بر پایه توالی از نقاط گره‌ای t_i بنا نهاده شده‌است. مطابق با فرمول دبور [23]:

$$f(t) = \sum_{i=1}^{n} c_i \, N_{i,d}(t) \tag{30}$$

که در آن

$$N_{i,r}(t) = \frac{t - t_i}{t_{i+r} - t_i} N_{i,r-1}(t) + \frac{t_{i+r+1} - t}{t_{i+r+1} - t_{i+1}} N_{i+1,r-1}(t)$$
$$r = 1, \dots, d \tag{31}$$

$$N_{i,0}(t) = \begin{cases} 1, & t_i \le t < t_{i+1}, \\ 0, & t < t_i \text{ یا } t \ge t_{i+1}, \\ 0, & t_i = t_{i+1} \end{cases} \tag{32}$$

که در معادله (30)، n تعداد نقاط کنترلی می‌باشد. تعریف معادله (32) کمک می‌کند تا زمانی که $t_i = t_{i+1}$ می‌شود، مخرج کسر در معادله (31) صفر نشده و این معادله نامعین نگردد. طرز انتخاب توالی نقاط گره‌ای شکل توابع پایه‌ای و در نتیجه شکل منحنی را تعیین می‌کند. $U = \{t_0, \dots, t_{m-1}\}$، بردار نقاط گره‌ای در نظر گرفته می‌شود. با لحاظ شرط صعودی بودن در بردار نقاط گره‌ای، اگر اولین و آخرین گره $d + 1$ بار تکرار شوند به بردار نقاط گره‌ای، غیریکنواخت گویند. در چنین حالتی منحنی از نقاط کنترلی اولیه و پایانی عبور خواهد کرد. با این تعریف تعداد نقاط گره‌ای برابر $m = n + d + 1$ می‌باشد [24].

برای مسئله پیش رو در این مقاله، کاربر نقاط مورد نظر خود را تعیین می‌نماید و برنامه‌ی بهینه‌سازی، 2 نقطه‌ی دیگر بعد از موقعیت اول و قبل از موقعیت آخر را به نحوی تعیین می‌نماید تا شرایط مرزی تحقق یابند. تعداد نقاط تعیین شده توسط کاربر با v_p نمایش داده می‌شود. با دو نقطه‌ی مجازی که فرآیند حل ایجاد می‌کند، $v_p + 2$ نقطه در تعیین منحنی بی‌اسپیلاین موجود است. همان‌طور که بیان شد برای عبور منحنی بی‌اسپیلاین از نقاط ابتدایی و انتهایی بایستی در بردار گره $d + 1$ نقطه گره‌ای یکسان در ابتدا و $d + 1$ نقطه گره‌ای یکسان در انتها موجود باشد. مطابق با $v_p + 2$ نقطه مورد نظر، $v_p + 2$ نقطه‌ی گره‌ای در بردار گره موجود است و با شرط بیان‌شده، d نقطه در ابتدا و d نقطه در انتها به آن اضافه می‌شود. بنابراین تعداد نقاط گره‌ای مطابق رابطه‌ی (33) محاسبه می‌گردد.

$$m = (v_p + 2) + 2d = (v_p + 2) + 10 = v_p + 12 \tag{33}$$

نقاط گره‌ای در این مسئله نقش زمان را ایفا می‌کنند. با توجه به رابطه‌ی $m = n + d + 1$، تعداد نقاط گره‌ای$(v_p + 12)$ و درجه‌ی 5 بودن منحنی، تعداد نقاط کنترلی موقعیت برابر $v_p + 6$ می‌باشد. برای موقعیت زوایای مفاصل با نقاط کنترلی $CPQ_i, (i = 1, \dots, n)$ و نقاط گره‌ای $Uq_i, (i = 1, \dots, m)$ منحنی موقعیت بصورت رابطه (34) است.

$$q(t) = \sum_{i=1}^{n} CPQ_i \, N_{i,d}(t) \tag{34}$$

وقتی منحنی موقعیت از درجه 5 است، منحنی سرعت از درجه 4 می‌باشد و تکرار نقاط گره‌ای در دو انتها از 6 به 5 نقطه کاهش می‌یابد پس در مجموع برای سرعت دو نقطه‌ی گره‌ای کمتر خواهیم داشت. برای منحنی سرعت با نقاط کنترلی $CPV_i, (i = 1, \dots, n-1)$ و نقاط گره‌ای $Uv_i, (i = 1, \dots, m-2)$ نقاط کنترلی سرعت بصورت رابطه (35) تعریف می‌شوند [10].

شکل 3 شکل (الف) نحوه‌ی تعیین نقاط میانی مسیر به کمک سنسور کینکت در شروع و انتهای مسیر و شکل (ب) تعیین نقاط میانی مسیر در سایر نقاط مسیر را نمایش می‌دهند.

$$CPV_i = \frac{d}{Uq_{i+d+1} - Uq_{i+1}} (CPQ_{i+1} - CPQ_i) , \quad i = 1, \dots, n-1 \tag{35}$$

بنابراین منحنی سرعت بصورت (36) می‌باشد.

$$v(t) = \sum_{i=1}^{n-1} CPV_i \, N_{i,d-1}(t) \tag{36}$$

و شرایط مرزی سرعت بصورت (37) و (38) خواهند بود.

$$CPV_1 = v_{\text{in}} \tag{37}$$

$$CPV_{n-1} = v_{\text{f}} \tag{38}$$

حال برای منحنی شتاب، نقاط کنترلی شتاب بصورت $CPA_i \; (i = 1, \dots, n-2)$ و نقاط گره‌ای آن بصورت $Ua_i, (i = 1, \dots, m-4)$ تعریف می‌شود.

$$CPA_i = \frac{d-1}{Uv_{i+d} - Uv_{i+1}} (CPV_{i+1} - CPV_i) , \quad i = 1, \dots, n-2 \tag{39}$$

و منحنی شتاب به صورت رابطه (40) می‌باشد.

$$a(t) = \sum_{i=1}^{n-2} CPA_i \, N_{i,d-2}(t) \tag{40}$$

و برای شرایط مرزی

$$CPA_1 = a_{\text{in}} \tag{41}$$

$$CPA_{n-2} = a_{\text{f}} \tag{42}$$

در انتها منحنی جرک مشخص می‌شود. برای منحنی جرک، نقاط کنترلی جرک بصورت $CPJ_i, (i = 1, \dots, n-3)$ و نقاط گره‌ای آن بصورت $Uj_i, (i = 1, \dots, m-6)$ تعریف می‌شود.

$$CPJ_i = \frac{d-2}{Ua_{i+d-1} - Ua_{i+1}} (CPA_{i+1} - CPA_i) , \quad i = 1, \dots, n-3 \tag{43}$$

و منحنی جرک به صورت رابطه (44) می‌باشد.

$$j(t) = \sum_{i=1}^{n-3} CPJ_i \, N_{i,d-3}(t) \tag{44}$$

و برای شرایط مرزی

$$CPJ_1 = j_{\text{in}} \tag{45}$$

$$CPJ_{n-3} = j_{\text{f}} \tag{46}$$

فرض می‌شود ربات در ابتدا و انتهای حرکت در حالت سکون قرار دارد، بنابراین $v_{\text{in}} = v_{\text{f}} = a_{\text{in}} = a_{\text{f}} = j_{\text{in}} = j_{\text{f}} = 0$ می‌باشند. حال v_{p} معادله را به گونه‌ای در نظر می‌گیریم که منحنی تشکیل شده از v_{p} نقطه‌ی میانی مورد نظر عبور کند. اگر نقاط تعیین شده توسط کاربر با $VR_i \; , \; i = 1, \dots, v_{\text{p}}$ نشان داده شوند، v_{p} معادله بصورت (47) خواهند بود.

$$\sum_{k=1}^{v_{\text{p}}+6} CPQ_k \, N_{k,d}(\tau_i) = VR_i \quad , i = 1, \dots, v_{\text{p}} \tag{47}$$

با در نظر گرفتن معادلات (35)، (37)، (38)، (39)، (41)، (42)، (43)، (45)، (46) و (47)، دستگاه $v_{\text{p}} + 6$ معادله و $v_{\text{p}} + 6$ مجهول رابطه‌ی (48) حاصل می‌شود. برای هر مفصل بازوی رباتیکی این دستگاه تکرار شده و از هر کدام $v_{\text{p}} + 6$ نقطه‌ی کنترلی بدست می‌آید که به وسیله‌ی آن‌ها منحنی زوایای مفاصل قابل ترسیم می‌باشند.

$$\begin{gathered}
-\frac{d}{Uq_{d+2} - Uq_2} CPQ_1 + \frac{d}{Uq_{d+2} - Uq_2} CPQ_2 = v_{\text{in}} \\
k_1 CPQ_1 + k_2 CPQ_2 + k_3 CPQ_3 = a_{\text{in}} \\
e_1 CPQ_1 + e_2 CPQ_2 + e_3 CPQ_3 + e_4 CPQ_4 = j_{\text{in}} \\
\sum_{k=1}^{v_{\text{p}}+6} CPQ_k \, N_{k,d}(\tau_i) = VR_i \quad , i = 1, \dots, v_{\text{p}} \\
-\frac{d}{Uq_{n+d} - Uq_n} CPQ_{n-1} + \frac{d}{Uq_{n+d} - Uq_n} CPQ_n = v_{\text{f}} \\
k_{n-2} CPQ_{n-2} + k_{n-1} CPQ_{n-1} + k_{n-1} CPQ_n = a_{\text{f}} \\
e_{n-3} CPQ_{n-3} + e_{n-2} CPQ_{n-2} + e_{n-1} CPQ_{n-1} + e_{n-1} CPQ_n = j_{\text{f}}
\end{gathered} \tag{48}$$

که در آن ضرایب $e_i, k_i, (i = 1,2, \dots, n)$ ضرایب نقاط کنترلی موقعیت می‌باشند. حال دوباره معادله بهینه‌سازی (2) را مورد بررسی قرار می‌دهیم. برای تحقق محدودیت‌های سرعت، شتاب و جرک کافیست نقاط کنترلی آن‌ها در این محدودیت صادق باشند [10].

$$|CPV_i| \le VC_i , \quad |CPA_i| \le AC_i, \quad |CPJ_i| \le JC_i \quad i = 1, \dots, N \tag{49}$$

3-4- تعیین بردار تقسیم زمان

برای شروع فرآیند بهینه‌سازی بایستی بردار اولیه‌ای وارد شود که به آن، بردار اولیه تقسیم زمان اطلاق می‌گردد.

$$ti_0 = \left(\tau_{01}, \tau_{02}, \dots, \tau_{0(v_{\text{p}}-1)}\right) \tag{50}$$

که در آن $\tau_{0j}, (j = 1,2, \dots, v_{\text{p}} - 1)$ حدس زمان مورد نیاز اولیه برای قطعه‌ی jام می‌باشد. در این مقاله متناسب با فاصله‌ی دو نقطه، زمان اولیه خاصی به صورت خودکار به آن قسمت از مسیر تعلق می‌گیرد. هرچه فاصله دو نقطه از یک قسمت از مسیر بیشتر باشد زمان اختصاصی به آن نیز بیشتر می‌شود. باید توجه داشت که

$$\tau_{01} + \tau_{02} + \dots + \tau_{0(v_{\text{p}}-1)} = T_{\text{d}} \tag{51}$$

که در آن T_{d} زمان کل انجام عملیات می‌باشد. در این مقاله از تابع بهینه‌سازی موجود در نرم افزار متلب برای حل مسئله بهینه‌سازی استفاده می‌شود. خروجی، زمان قطعه‌های مسیر و نقاط کنترل مطلوب است. بردار تقسیم زمان بصورت (52) می‌باشد.

$$ti = \left(\tau_1, \tau_2, \dots, \tau_{(v_{\text{p}}-1)}\right), \tau_1 + \tau_2 + \dots + \tau_{(v_{\text{p}}-1)} = T_{\text{d}} \tag{52}$$

که در آن $\tau_j, (j = 1,2, \dots, v_{\text{p}} - 1)$ زمان مورد نیاز نهایی برای قطعه‌ی jام مسیر می‌باشد. بنابراین شرط معادله (52) نیز باید در مسئله بهینه‌سازی لحاظ گردد. با این بردار می‌توان بردار نقاط گره‌ای را تشکیل داد و با در اختیار داشتن نقاط کنترلی به همراه آن منحنی‌های موقعیت، سرعت، شتاب و جرک قابل رسم می‌باشند.

4- الگوریتم عدم برخورد با مانع

1-4- چگونگی تعیین مانع

در این مقاله فرض بر این است که موقعیت نقاط گوشه‌ای اشیا و موانع به گونه‌ای توسط الگوریتم‌های بینایی ماشین تعیین گردیده و در اختیار قرار گرفته‌است. به عنوان مثال نقاط گوشه‌ای جسمی خاص در شکل 4 نشان داده شده‌است. با استفاده از دستگاه مختصات متصل به دوربین و دستگاه مختصات مرجع ربات و انتقال دستگاه دوربین نسبت به دستگاه ربات موقعیت این نقاط حاصل می‌شوند. بعد از مفروض بودن نقاط گوشه‌ای، برای اجرای گذر از مانع روشی در نظر گرفته‌شده، بدین صورت که ماکزیمم و مینیمم مولفه‌های موقعیت (x, y, z) بدست آمده و مانع به صورت مکعبی در نظر گرفته می‌شود به صورتی که ابعاد آن مطابق (53) می‌باشد

$$x_{\min} - \beta \le \text{طول} \le x_{\max} + \beta$$

$$y_{\min} - \gamma \le \text{عرض} \le y_{\max} + \gamma$$

$$z_{\min} - \varepsilon \leq \text{ارتفاع} \leq z_{\max} + \varepsilon \quad (53)$$

که در آن β، γ و ε مقادیری ثابت برای حاشیه امنیت می‌باشند. در مرحله‌ی نقاطی به نام نقاط بحرانی بر روی لینک‌ها تعیین می‌گردند. بدین منظور ابتدا بر روی مانع نقاطی را تعیین کرده تا هر لینک در نظر گرفته‌شده نسبت به آن‌ها سنجیده شوند. شکل 5 برای هر مانع که بصورت مکعب در نظر گرفته‌شده، نقاط درنظرگرفته‌شده بر روی مانع را نشان می‌دهد. همان‌طور که مشاهده می‌شود در مجموع 20 نقطه مشخص شده که 8 نقطه در 8 کنج قرار داشته و 12 نقطه‌ی دیگر در وسط هر یال قرار می‌گیرد.

4-2- نقاط بحرانی هر لینک از بازوی رباتیکی

نقطه‌ی بحرانی به نقطه‌ای در امتداد لینک گفته می‌شود که نسبت به مانع در نزدیک‌ترین موقعیت قرار می‌گیرد. اگر دو سر هر لینک با M و N مشخص شود و نقطه مانعی با O نشان داده‌شود، یکی از سه حالت شکل 6 برای نقاط بحرانی ایجاد می‌شود [19]. با توجه به آن‌که نزدیک‌ترین فاصله‌ی یک نقطه نسبت به یک خط برابر با اندازه‌ی پاره‌خطی عمود بر خط از آن نقطه است و این مقدار در شکل 6 با OO_{L} برابر می‌باشد. می‌توان نوشت

$$OO_{\mathrm{L}}{}^2 = OM^2 - O_{\mathrm{L}}M^2 = ON^2 - O_{\mathrm{L}}N^2 \quad (54)$$

طول هر لینک با $NM = L > 0$ نشان داده می‌شود. بنابراین داریم

$$NO_{\mathrm{L}} = NM - O_{\mathrm{L}}M \quad (55)$$

بنابراین

$$OM^2 - O_{\mathrm{L}}M^2 = ON^2 - (NM - O_{\mathrm{L}}M)^2 \quad (56)$$

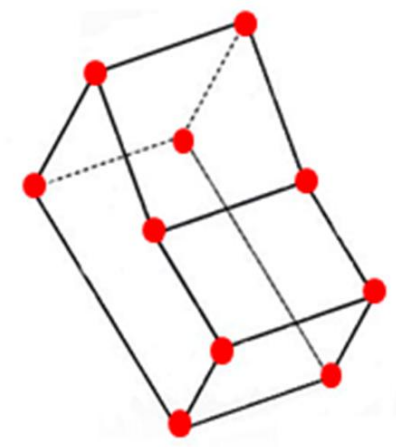
در نتیجه

$$O_{\mathrm{L}}M = (OM^2 + NM^2 - ON^2)/(2NM) \quad (57)$$

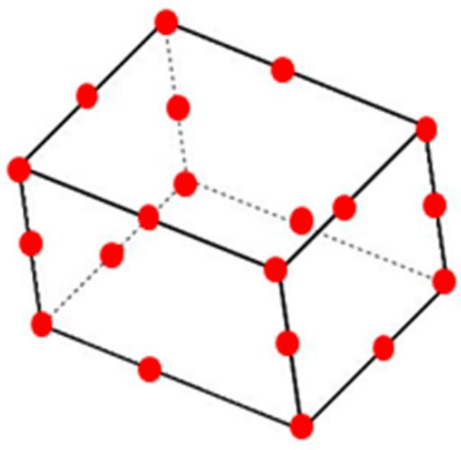
حال 3 حالت می‌توان در نظر گرفت.

حالت $a-$ اگر $0 < O_{\mathrm{L}}M < L$ باشد نقطه‌ی بحرانی C همان O_{L} می‌باشد و نزدیک‌ترین فاصله به مانع (OC) برابر OO_{L} است.

حالت $b-$ اگر $O_{\mathrm{L}}M \geq L$ باشد نقطه‌ی بحرانی C همان N می‌باشد و نزدیک‌ترین فاصله به مانع (OC) برابر ON است.



شکل 4 نقاط گوشه‌ای اشکال سه بعدی



شکل 5 نقاط در نظر گرفته‌شده بر روی مانع برای عدم برخورد

حالت $c-$ اگر $0 \leq O_{\mathrm{L}}M$ باشد نقطه‌ی بحرانی C همان M می‌باشد و نزدیک‌ترین فاصله به مانع (OC) برابر OM است.

3 طول کلی را می‌توان برای برخورد با مانع در نظر گرفت که در شکل 7 این سه طول با N_1M_1، N_2M_2 و N_3M_3 نمایش داده شده‌اند.

این سه طول به صورت زیر تعریف می‌گردند.

N_1M_1- از مرکز دستگاه دوم تا مرکز دستگاه سوم

N_2M_2- از مرکز دستگاه چهارم تا مرکز دستگاه پنجم

N_3M_3- از مرکز دستگاه ششم تا مرکز دستگاه هشتم

با توجه به بزرگترین قطری که هر قسمت از این طول‌ها دارند محدوده‌ای برای OC می‌توان تعیین کرد که به صورت زیر می‌باشد.

برای N_1M_1، N_2M_2 و N_3M_3 به ترتیب در رابطه (58)،(59) و (60) داریم:

$$OC > 0.11\text{m} \quad (58)$$

$$OC > 0.11\text{m} \quad (59)$$

$$OC > 0.07\text{m} \quad (60)$$

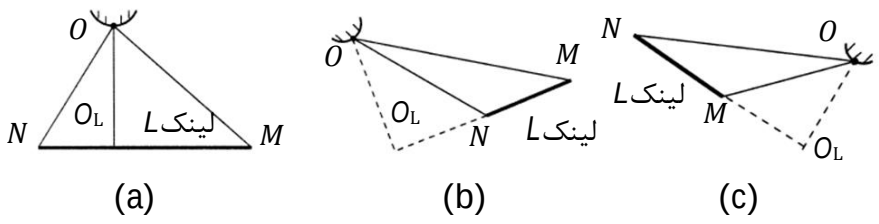
4-3- بیان الگوریتم گذر از مانع

همان‌طور که در بخش سینماتیک معکوس بیان شده‌است زاویه‌ی θ_1 با سینماتیک معکوس برابر با $\mathrm{atan2}(p_y, p_x) - \alpha$ بدست می‌آمد. مقدار α به گونه‌ای بود که $-40 > \alpha > 40$ و ربات بیشترین قابلیت حرکت را داشته‌باشد. برای گذر از مانع باید تعریفی در نظر گرفته شود تا علاوه بر ویژگی‌های بیان شده، فاصله‌ی مطمئن تا لینک‌های ربات نیز حاصل شود. بنابراین سینماتیک معکوس به صورت زیر با تغییر کوچکی همراه است.

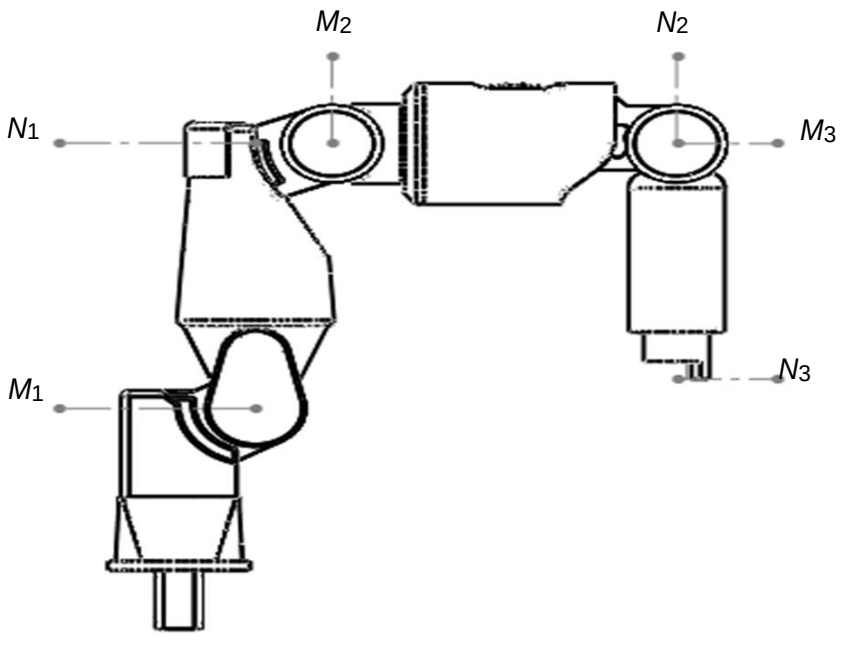
زاویه α به صورتی تعیین می‌شود که

$$-40 < \alpha < 40$$

زاویه α به صورتی تعیین می‌شود که هیچ کدام از طول‌های در نظر گرفته‌شده با مانع برخوردی نداشته باشند. در واقع در شروطی که در قسمت قبل بیان شد صدق کنند.



شکل 6 حالات ممکن برای تعیین نقطه بحرانی (O نقطه‌ی مانعی و NM طول لینک مورد بررسی است) [19]



شکل 7 طول‌های حساس ربات برای الگوریتم عدم برخورد با مانع

زاویه α به صورتی تعیین می‌شود که روند تغییرات زوایای طراحی شده در فضای مفصلی در حالت جدید به روند سابق نزدیک باشد.

با این سه شرط سینماتیک معکوسی در اختیار داریم که کل مسیر را می‌توان به صورتی در فضای مفصلی تغییر داد تا نتیجه‌ی مطلوب یعنی گذر از نقاط تعیین شده به گونه‌ای انجام شود که برخوردی با مانع به وجود نیاید. فلوچارت الگوریتم در نظر گرفته‌شده بصورت شکل 8 می‌باشد.

5- نتایج شبیه‌سازی

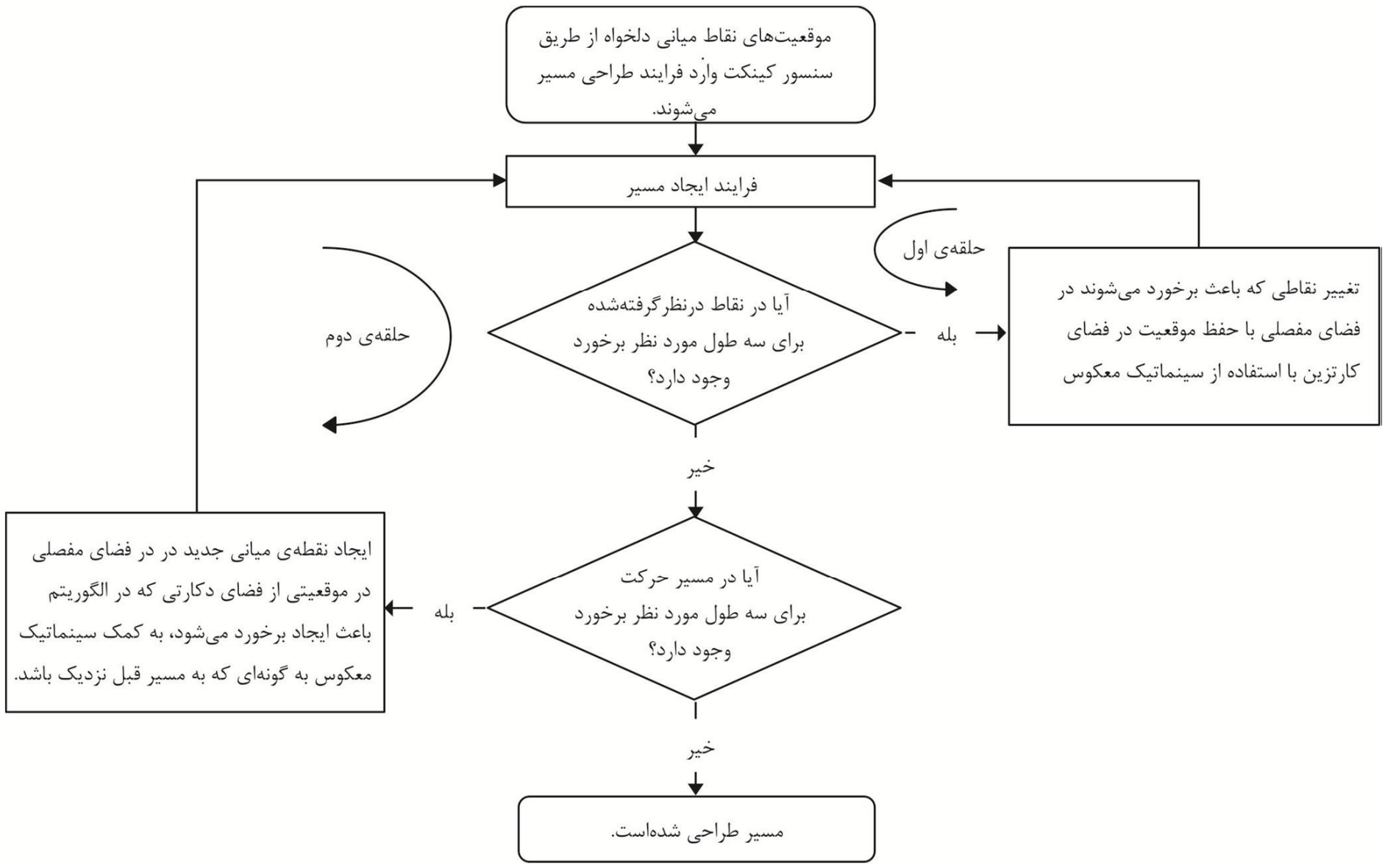
در سناریوی در نظر گرفته‌شده در این شبیه‌سازی نقاط میانی اولیه تعیین‌شده توسط کاربر به وسیله‌ی سنسور کینکت در فضای کارتزین مطابق جدول 3 می‌باشند.

نحوه‌ی تعیین این نقاط در شکل 9 نشان داده شده‌است. موقعیت این نقاط در فضای مفصلی به کمک سینماتیک معکوس در جدول 4 آورده شده‌است. این حرکت بایستی در مدت زمان 3 ثانیه انجام شود به صورتی که سرعت از محدوده 100 درجه بر ثانیه عدول نکند. مسیر طراحی‌شده‌ی اولیه در فضای کارتزین مطابق الگوریتم بهینه‌سازی ارائه‌شده در این مقاله، در شکل 10 نشان داده شده و ربات حرکتی مطابق با شکل 11 را به انجام می‌رساند. شبیه‌سازی حرکت سه‌بعدی ربات در محیط سیممکانیک انجام گردیده‌است.

حال فرض می‌شود نقاط ماکزیمم و میینمم ابعاد مانع به صورت رابطه‌ی (61) باشد.

جدول 3 نقاط میانی اولیه برای طراحی مسیر در فضای کارتزین

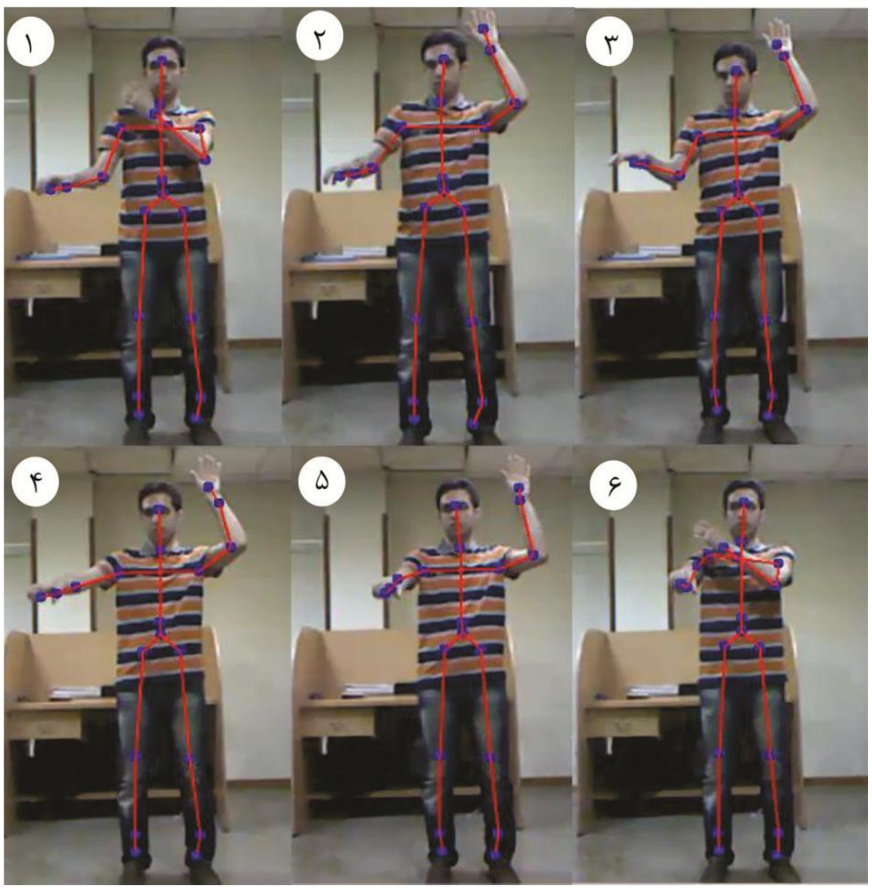
شماره‌ی نقطه	x(m)	y(m)	z(m)
نقطه اول	0/3601	0/4121	0/0301
نقطه دوم	0/3602	0/4122	0/0756
نقطه سوم	0/3203	0/4201	0/1503
نقطه چهارم	0/5021	0/4331	0/1543
نقطه پنجم	0/5111	0/2903	0/1774
نقطه ششم	0/5211	0/2256	0/1905



شکل 8 فلوچارت عدم برخورد با مانع

جدول 4 نقاط میانی اولیه برای طراحی مسیر در فضای مفصلی

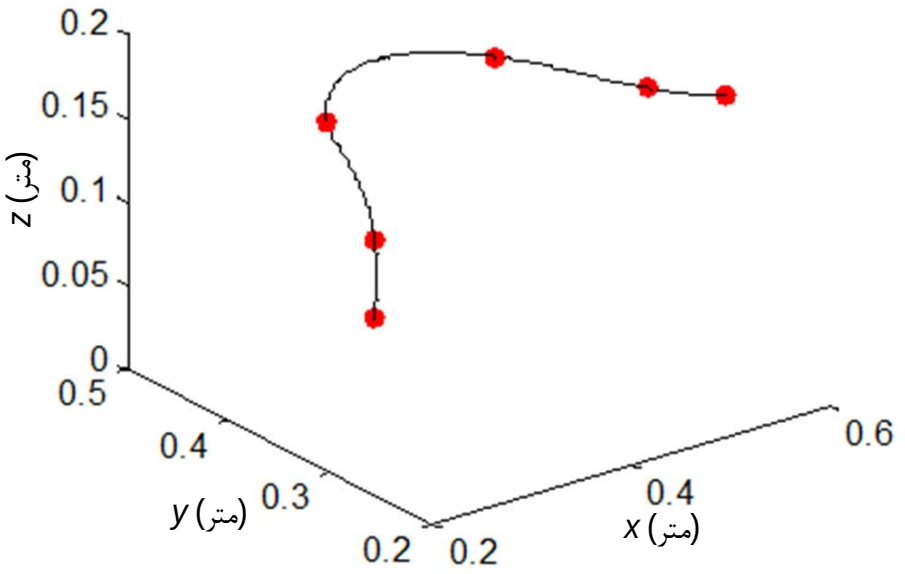
شماره نقطه	θ_1(درجه)	θ_2(درجه)	θ_3(درجه)	θ_4(درجه)	θ_5(درجه)	θ_6(درجه)	θ_7(درجه)
نقطه اول	28/9	-12/8	20/6	105/8	4/5	86/2	0
نقطه دوم	28/9	-12/9	20/2	100	4/4	92	5/1
نقطه سوم	32/7	-14/2	19/2	91/7	4/7	101/7	11/5
نقطه چهارم	20/8	4/9	25/6	70/9	-2/2	104/6	20/6
نقطه پنجم	9/6	-5/3	21/8	79/3	2	105/6	25/6
نقطه ششم	3/4	-7/5	21	79/9	2/8	107/1	32



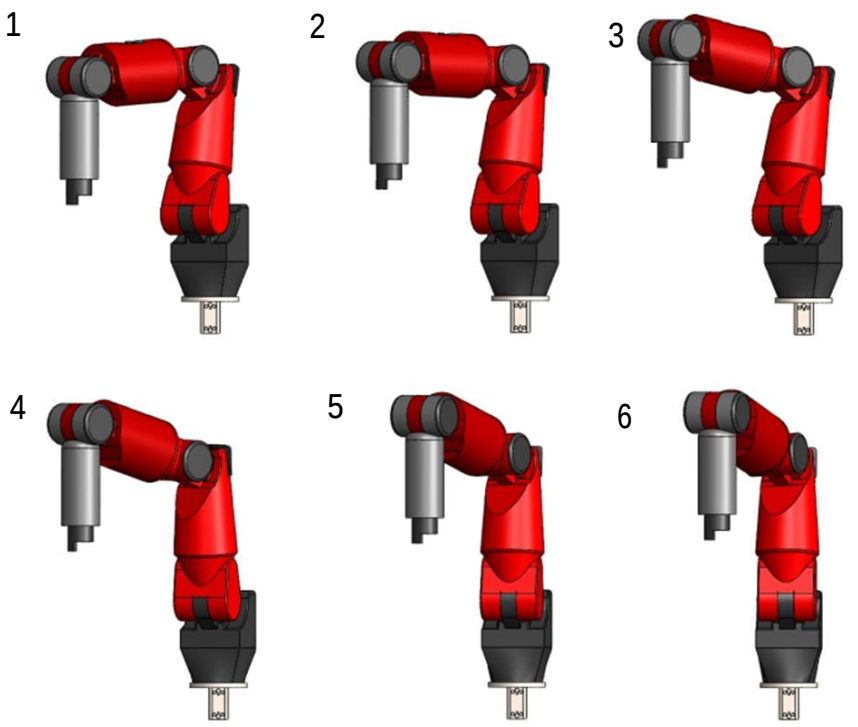
شکل 9 تعیین نقاط میانی اولیه توسط کاربر

همانند شکل 5، بیست نقطه‌ی مانعی تولید می‌شود. پس از ورود به الگوریتم، در پاسخ به این سوال که آیا نقاط تعیین شده در وضعیت برخورد با مانع قرار دارند یا خیر؟ جواب بله است. در نقطه‌ی چهارم برخورد موجود است (این برخورد در شکل 12 نشان داده شده‌است).

بنابراین الگوریتم وارد حلقه‌ی اول از شکل 8 شده و این نقطه مطابق جدول 5 تغییر می‌یابد. با توجه به افزونگی ربات نقطه میانی اگرچه در فضای مفصلی تغییر یافته، در محیط کارتزین تغییری پیدا نمی‌کند. شکل 13 نقطه‌ی چهارم را بعد از تغییر نشان می‌دهد. بعد از ایجاد دوباره‌ی مسیر، دیگر نقاط تعیین‌شده در منطقه‌ی برخورد با مانع قرار نمی‌گیرند. سپس طول مسیر برای وجود مانع بررسی می‌شود. دیده می‌شود بین نقطه‌ی چهارم و پنجم در طول مسیر نقطه‌ای در موقعیت $\begin{bmatrix}0.5209\\0.4126\\0.1542\end{bmatrix}$ وجود دارد که در آن برخورد با مانع ایجاد می‌شود. بنابراین الگوریتم وارد حلقه‌ی دوم از شکل 8 شده و این نقطه در فضای کارتزین به عنوان نقطه‌ی جدید میانی تعیین گردیده و در فضای مفصلی به صورتی تعیین می‌شود که برخورد با مانع صورت نگیرد. در این حالت نقاط میانی مطابق جدول 6 می‌باشند. در این حالت پس از بازگشت به ابتدای الگوریتم و طراحی مسیر، نقاط میانی تعیین‌شده مطابق انتظار در منطقه‌ی برخورد با مانع قرار نداشته و بعد از بررسی دوباره مسیر مشاهده می‌شود پس از تعیین نقطه‌ی میانی هفتم در طول مسیر نیز برخوردی موجود نیست. مسیری که در فضای کارتزین مجری نهایی ربات طی کرده در شکل 14 نشان داده شده‌است. با توجه به اعمال الگوریتم عدم برخورد با مانع در شکل 14 نشان داده شده که مسیر نهایی به مسیر اولیه اعمالی در شکل 10، توسط کاربر نزدیک می‌باشد.

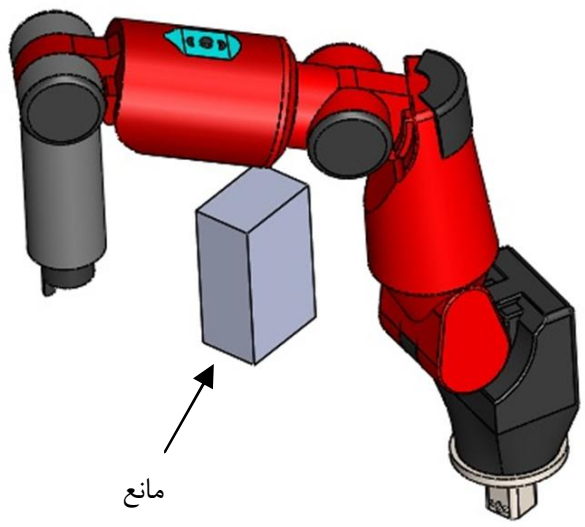


شکل 10 مسیر طراحی شده قبل از ورود به الگوریتم عدم برخورد با مانع

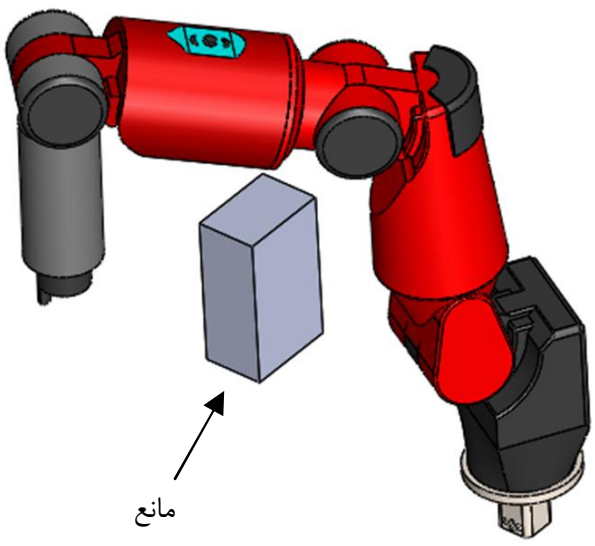


شکل 11 حرکت ربات در مسیر طراحی شده‌ی اولیه مطابق با نقاط اولیه میانی

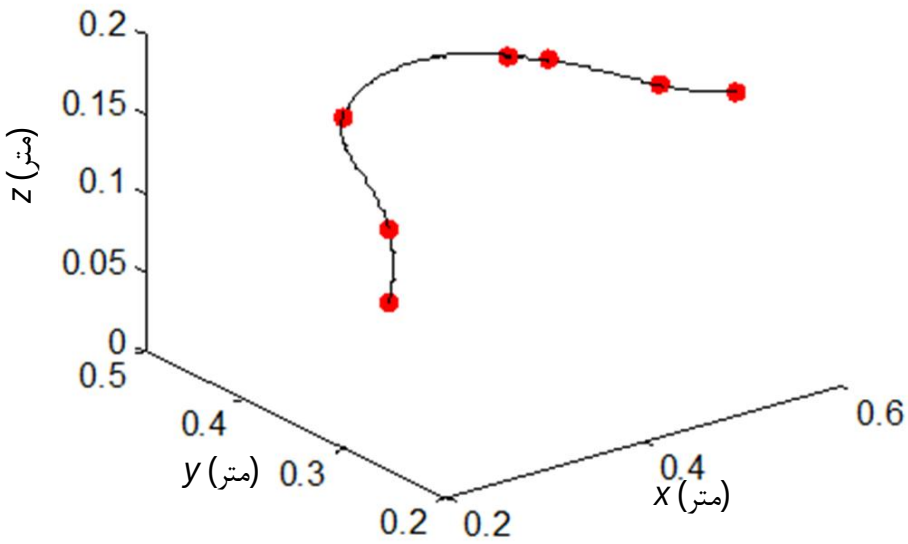
$$\begin{aligned}0.25\text{m} \le x_{\text{obs}} \le 0.4\text{m}\\ 0\text{m} \le y_{\text{obs}} \le 0.1\text{m}\\ 0.1\text{m} \le z_{\text{obs}} \le 0.35\text{m}\end{aligned} \quad (61)$$



شکل 12 موقعیت بازو در نقطه‌ی چهارم و برخورد آن با مانع در طول N_2M_2



شکل 13 موقعیت بازو در نقطه‌ی چهارم پس از تغییر موقعیت در فضای مفصلی

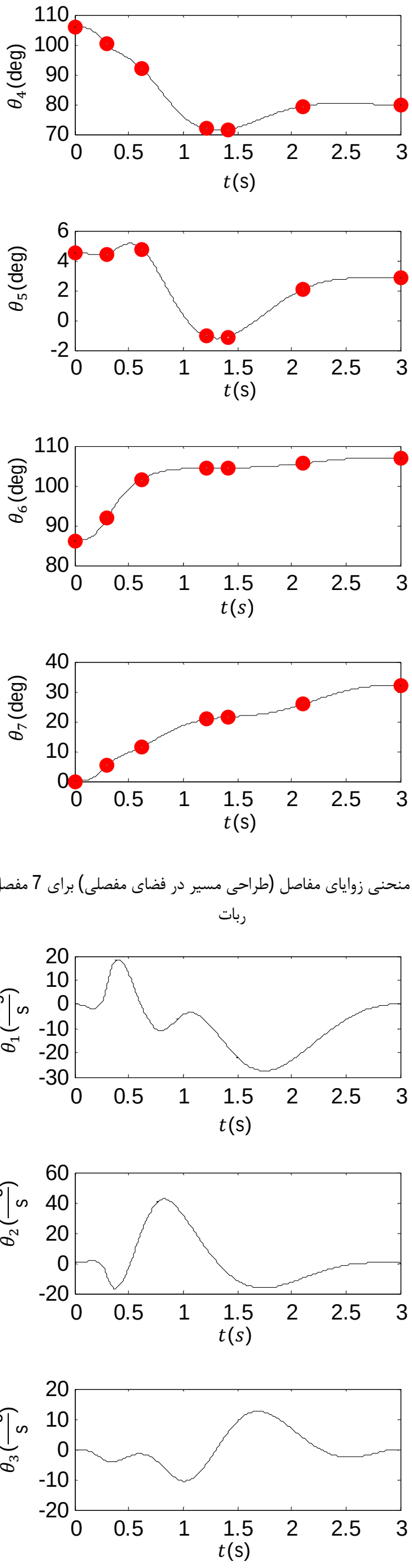


شکل 14 مسیر طراحی‌شده نهایی بعد از اعمال الگوریتم عدم برخورد با مانع

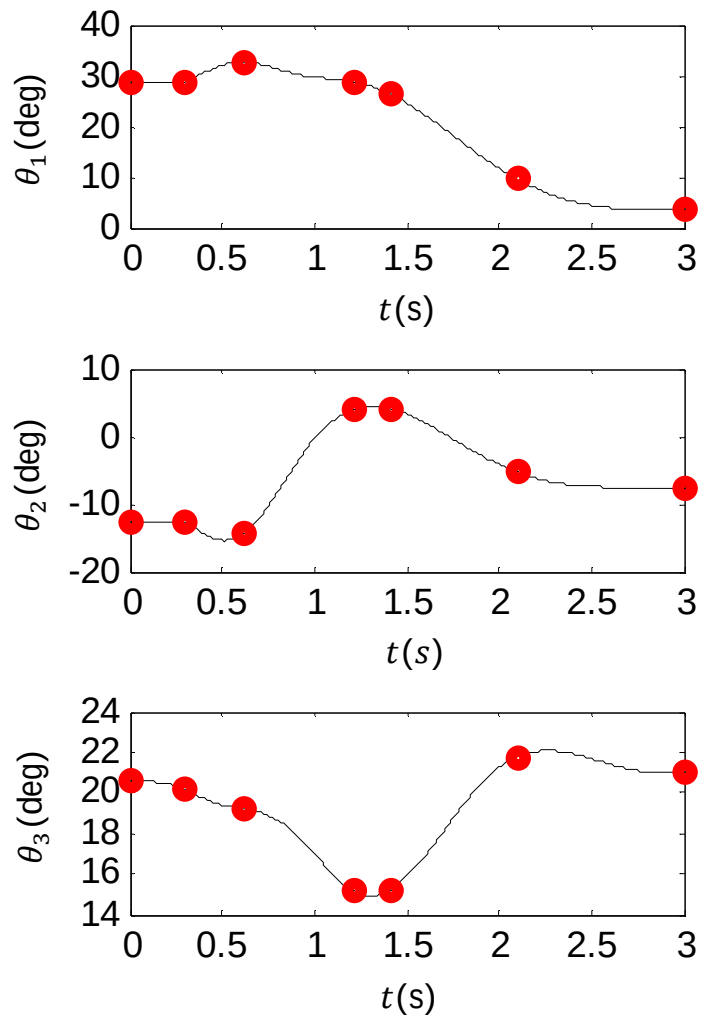
پس از تعیین نقطه‌ی هفتم و ورود بردار تقسیم زمانی رابطه‌ی (50) مطابق با آنچه در بخش 3-4 بیان گردید، انتگرال رابطه‌ی (28) برابر با $5.3153 \times 10^5 \left(\frac{m^2}{s^6}\right)$ شده و پس از بهینه‌سازی و مینیمم گشتن، این مقدار به $3.1011 \times 10^5 \left(\frac{m^2}{s^6}\right)$ رسیده‌است. منحنی زوایای مفاصل در شکل 15 و منحنی سرعت‌های زاویه‌ای در شکل 16 نمایش داده شده‌است که نشان می‌دهند محدوده‌ی سینماتیکی برای سرعت زاویه‌ای (100 درجه بر ثانیه) رعایت شده‌است.

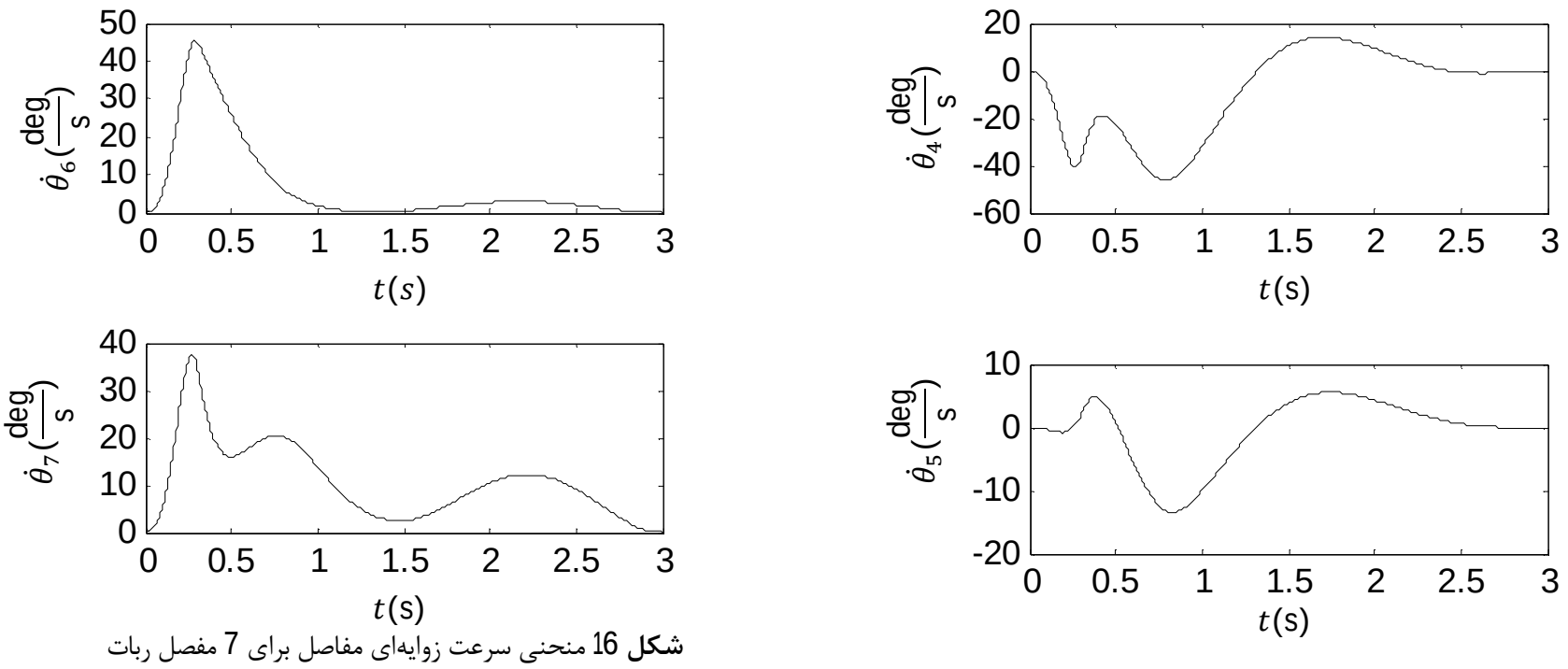
6- نتیجه‌گیری

در این مقاله روشی جدید برای طراحی مسیر بازوی رباتیکی هفت درجه آزادی ارائه شده‌است. مطابق با این روش، کاربر ربات چند نقطه‌ی میانی از مسیر موردنظر را توسط دوربین کینکت به بازو انتقال داده و روش‌پیشنهادی مسیر مورد نظر را با تلفیق روش جرک- مینیمم و استفاده از توابع بی‌اسپیلاین و محدودیت‌های سینماتیکی ربات، بهینه نموده و اجرا می‌نماید. همچنین گذر از موانع استاتیک نیز در این روش پیشنهادی مورد ملاحظه قرار گرفته‌است. این روش می‌تواند به عنوان یک راهکار مناسب برای طراحی مسیر بازوهای رباتیکی صنعتی مورد استفاده قرار گیرد.



شکل 15 منحنی زوایای مفاصل (طراحی مسیر در فضای مفصلی) برای 7 مفصل ربات





شکل 16 منحنی سرعت زاویه‌ای مفاصل برای 7 مفصل ربات

جدول 5 نقطه‌ی میانی چهارم پس از تغییر مطابق الگوریتم عدم برخورد با مانع، در فضای مفصلی

شماره نقطه	θ_1 (درجه)	θ_2 (درجه)	θ_3 (درجه)	θ_4 (درجه)	θ_5 (درجه)	θ_6 (درجه)	θ_7(درجه)
نقطه چهارم	28/8	4	15/2	71/8	-1/1	104/4	20/6

جدول 6 نقاط میانی تعیین شده پس از بررسی درون الگوریتم عدم برخورد با مانع، در فضای مفصلی

شماره نقطه	θ_1 (درجه)	θ_2 (درجه)	θ_3 (درجه)	θ_4 (درجه)	θ_5 (درجه)	θ_6 (درجه)	θ_7(درجه)
نقطه اول	28/9	-12/8	20/6	105/8	4/5	86/2	0
نقطه دوم	28/9	-12/9	20/2	100	4/4	92	5/1
نقطه سوم	32/7	-14/2	19/2	91/7	4/7	101/7	11/5
نقطه چهارم	28/8	4	15/2	71/8	-1/1	104/4	20/6
نقطه پنجم	26/4	4/2	15/2	71/5	-1/1	104/5	21/4
نقطه ششم	9/6	-5/3	21/8	79/3	2	105/6	25/6
نقطه هفتم	3/4	-7/5	21	79/9	2/8	107/1	32

7- مراجع

[1] R. Paul and H. Zhang, "Robot motion trajectory specification and generation," in *Robotics Research: The Second International Symposium,* vol. 3, Kyoto, 1985, pp. 373–380.

[2] A. J. Koivo, *Fundamentals for Control of Robotic Manipulators*.: John Wiley & Sons, 1989.

[3] R. E. Parkin, *Applied Robotic Analysis*.: Prentice Hall, 1991.

[4] J.J. Craig, *Introduction to Robotics: Mechanics and Control,* 3rd ed.: Pearson Prentice Hall, 2005.

[5] M.W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*.: John Wiley & Sons, 2006.

[6] R.N. Jazar, *Theory of Applied Robotics: Kinematics, Dynamics, and Control,* 2nd ed.: Springer, 2010.

[7] J. Angeles, *Fundamentals of Robotic Mechanical Systems,* 3rd ed.: Springer, 2007.

[8] C.-H. Wang and J.-G Horng, "Constrained minimum-time path planning for robot manipulators via virtual knots of the cubic B-spline functions," *IEEE Transactions on Automatic Control,* vol. 35, no. 5, pp. 573-577, May 1990.

[9] D. Constantinescu and E. A. Croft, "Smooth and time optimal trajectory planning for industrial manipulators along specified paths," *Journal of Robotic Systems,* vol. 17, no. 5, pp. 233-249, 2006.

[10] A. Gasparetto and V. Zanotto, "A new method for smooth trajectory planning of robot manipulators," *Mechanism and machine theory,* vol. 42 , no. 4, pp. 455–471, 2007.

[11] P. Huang, Y. Xu, and B. Liang, "Global minimum-jerk trajectory planning of space manipulator," *International Journal of Advanced Robotic Systems,* vol. 4, no. 4, pp. 405-413, 2006.

[12] R. Glasius, A. Komoda, and S. C. A. M. Gielen, "Neural network dynamics for path planning and obstacle avoidance," *Neural Networks,* vol. 8, no. 1, pp. 125 -133, 1995.

[13] S. X. Yang and M. Meng. "Real-time collision-free path planning of robot manipulators using neural network approaches," *Autonomous Robots,* vol. 9, no. 1, pp. 27-39, 2000.

[14] J.-J. Park, J.-H. Kim, and J.-B. Song, "Path Planning for a robot manipulator based on probabilistic roadmap and reinforcement learning," *International Journal of Control, Automation, and Systems,* vol. 5, no. 6, pp. 674-680, 2007.

[15] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The international journal of robotics research,* vol. 4, no. 3, pp. 109-117, 1985.

[16] M. Pourazady and L. Ho, "Collision avoidance control of redundant manipulators," *Mechanism and machine theory,* vol. 26, no. 6, pp. 603-611, 1991.

[17] K. Glass, R. Colbaugh, D. Lim, and H. Seraji, "Real-time collision avoidance for redundant manipulators," *Robotics and Automation, IEEE Transactions,* vol. 11, no. 3, pp. 448-457, 1995

[18] B. Daachi, A. Benallegue, T. Madani, and M. E. Daachi, "Neural networks for redundant robot manipulators control with obstacles avoidance," *Journal of Robotics and Mechatronics,* vol. 16, no. 1, pp. 90-96, 2004.

[19] Y. Zhang and J. Wang, "Obstacle avoidance for kinematically redundant manipulators using a dual neural network," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions,* vol. 34, no. 5, pp. 2126-2132, 2004.

[20] F. Najafi, M. Karimi, and M. Ghayour, "Optimal Trajectory Planning and Obstacle Avoidance of a Manipulator in the Presence of Ellipsoidal Obstacles Using Genetic Algorithms," *Mechanics, Journals of Modares,* vol. 10, no. 4. February 2011, pp. 75-84. (In Persian)

[21] D. L. Peiper, *The kinematics of manipulators under computer control* : Unpublished Ph.D Thesis, Stanford University, 1968.

[22] S. Tanaka et al., "Minimum-jerk trajectory generation for master-slave robotic system," in *Biomedical Robotics and Biomechatronics (BioRob), 2012 4th IEEE RAS & EMBS International Conference on. IEEE,* 2012, pp. 811-816.

[23] C. De Boor, *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.

[24] T. Lyche, K. Mørken, *Spline methods draft*. *Department of Informatics, University of Oslo,* Accessed 20 July 2014 http://heim. ifi. uio. no/knutm/komp04. pdf