The First National Conference on Software Engineering Applications
3-5 march 2009
اولین کنفرانس ملی
کاربردهای مهندسی نرم افزار لاهیجان

کاربرد نظریه گراف در حل مسائل برنامه ریزی حرکت روبات: گراف های قابل حل مینیمال

الیپس مسیحی[1]، هیوا صمدیان[2]
[1] دانشکده فنی و مهندسی دانشگاه تربیت مدرس، تهران، ایران.Masehian@modares.ac.ir
[2] دانشکده فنی و مهندسی دانشگاه تربیت مدرس، تهران، ایران.h.samadian@modares.ac.ir

چکیده

مسائل برنامه ریزی حرکت از جمله مباحث مطرح در حوزه روباتیک و هوش مصنوعی می باشند. روشهای متداول و سنتی در حل مسائل برنامه ریزی حرکت عبارت از نقشه راه، تجزیه سلولی، میدان پتانسیل، برنامه ریزی ریاضی، دیدنگار، نمودار ورونویی و چند روش دیگر ریاضی هستند. علی رغم اینکه بسیاری از این روشها در به دست آوردن خط سیر مناسب حرکت برای یک روبات موفق عمل می کنند، در حل مسئله برنامه ریزی حرکت چند روباتی کارایی ندارند. نظریه گراف، که به طور مستقیم در بسیاری از شاخه های فناوری اطلاعات مثل داده کاوی، وب کاوی و غیره به کار گرفته شده است، تاکنون به ندرت در حوزه روباتیک استفاده شده است. در این مقاله، نحوه بکارگیری نظریه گراف در مسائل برنامه ریزی حرکت و به خصوص برای مسایل چند روباتی ارائه خواهد شد. گراف های مینیمال قابل حل مفهومی جدید است که برای اولین بار در این مقاله معرفی شده است. خصوصیات این نوع گراف ها در قالب چند قضیه اثبات شده است.

کلمات کلیدی

نظریه گراف، برنامه ریزی حرکت چند روباتی، گراف های قابل حل، گراف های مینیمال، گراف های قابل حل مینیمال

۱ مقدمه

توسعه روزافزون روبات ها و گسترش استفاده از آنها در زمینه های مختلف، باعث بوجود آمدن فصلی نو در زندگی بشر شده است. با شروع هزاره سوم و ورود انسان به قرن بیست و یکم آرزوی انسان برای ساختن انسان ماشینی، دور از ذهن به نظر نمی رسد. این همه در سایه تحقیقات و مطالعات فراوانی است که در زمینه هوش مصنوعی و روباتیک انجام گرفته است و امروزه با به میدان آمدن ابزارهای محاسبات نرم[1] هوش مصنوعی و روباتیک وارد مرحله جدید از زندگی خود می شوند. رشد تکنولوژی در تمام زمینه ها به خصوص کامپیوتر و تکنولوژی های سخت افزاری نقش مهمی در پیشرفت فن آوری روباتیک داشته است بنابراین لازم است تا بخش های تئوری، الگوریتم ها و نرم افزار نیز تقویت و توسعه یابند.

یکی از موضوعات مطرح در بخش تئوری مربوط به روبات ها موضوع برنامه ریزی حرکت آنهاست. مسئاله *برنامه ریزی حرکت روبات*[2] عبارتست از یافتن مجموعه ای از حرکت ها که می تواند یک روبات را از یک وضعیت اولیه به یک موقعیت دیگر ببرد بدون اینکه تصادمی پیش بیاید [۱]. برنامه ریزی حرکت چند روباتی، پیدا کردن یک برنامه برای حرکت چند روبات است بطوریکه هر روبات به مقصد خود برسد بدون اینکه با سایر روبات ها و موانع محیط برخورد کند [۲]. برنامه ریزی حرکت شامل یافتن مسیرهایی در *فضای پیکربندی*[3] است [۳].

برنامه ریزی حرکت روبات بسته به پارامترهای مختلفی چون مقیاس، دامنه، کامل بودن[4]، منبع اطلاعات، در دسترس بودن اطلاعات، عدم اطمینان، ابعاد محیط، تغییرات زمانی محیط، تعداد روبات ها، نوع روبات ها، تغییر شکل روبات ها و محدودیت روبات ها، انواع مختلفی دارد ولی در هر حال به طور کلی شامل این اقدامات است: ۱- پیکربندی فضا، ۲- ارائه و نمایش روبات، ۳- برنامه ریزی مسیر[5]، ۴- تشخیص قابل حل بودن مسیر، ۵- بهینه کردن مسیر، ۶- اعمال قوانین حرکت. البته گاهی همه این اقدامات و گاهی فقط طراحی یک مسیر بهینه قابل حل و ارائه قوانین حرکت روبات ها، برنامه ریزی حرکت نامیده می شود. ما در اینجا طراحی یک مسیر بهینه قابل حل و ارائه قوانین حرکت، را به عنوان تعریف وظایف برنامه ریزی حرکت مورد نظر قرار داده ایم.

در این مقاله ابتدا به طور مختصر روشهای برنامه ریزی حرکت روبات و به ویژه روشهای مبتنی بر گراف در بخش ۲ معرفی می شوند. سپس، برنامه ریزی حرکت چند روباتی به عنوان نوعی از برنامه ریزی حرکت در بخش ۳ بیان می شود. در بخش ۴ کاربرد نظریه گراف در حل مساله برنامه ریزی حرکت چند روباتی طی چند مثال و قضیه تشریح خواهد شد. ضمن ارائه مفهوم گراف های قابل حل مینیمال برای اولین بار، قابلیت ها و خصوصیات آنها در حل مساله برنامه ریزی حرکت چند روباتی طی چند قضیه تحقیق می شود.

۲ روشهای حل مسأله برنامه ریزی حرکت

روشهای مختلفی برای برنامه ریزی حرکت وجود دارند که در اینجا به دو دسته روشهای مبتنی بر گراف و سایر روشها تقسیم بندی شده اند. روشهای مبتنی بر گراف برتری هایی نسبت به روشهای دیگر دارند که بعداً معرفی خواهند شد.

۲-۱ روشهای مبتنی بر گراف

۲-۱-۱ دیدنگار[6]

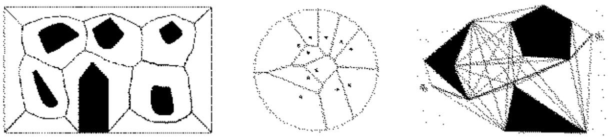
دیدنگار مجموعه خطوط در فضای آزاد است که یک خاصیت یک شی را به شی دیگر متصل می کند. در فرم اصلی اش، این خصوصیت، رئوس مانع چند گوشه ای می باشد. اگر n تعداد کل رئوس موانع باشد، در دیدنگار، $O(n^2)$ لبه وجود دارد که می توانند در زمان $O(n^2)$ و در دو بُعد ساخته شود. شکل ۱- (الف) دیدنگار چند گوشه ای را که رئوس به عنوان خصیصه ها هستند را نمایش میدهد. کوتاه ترین مسیر بین شروع و هدف از این نمودار محاسبه میشود.

تعمیمی از دیدنگار، *دیدنگار تعمیم یافته*[7] میباشد که موانع، چند گوشه ایی های تعمیم یافته هستند، یعنی نقاط می توانند هم روی گوشه های تیز باشند و هم واقع بر لبه های منحنی. اگر n تعداد کل رئوس موانع باشد، تعداد رئوس در دیدنگار تعمیم یافته $O(n^2)$ و تعداد یال ها نیز $O(n^2)$ میباشد. دیدنگار تعمیم یافته میتواند در زمان $O(n^3)$ ساخته شده و در زمان $O(n^2)$ جستجو شود. کوتاه ترین مسیر در زمان $O(n^2 \log n)$ با استفاده از الگوریتم *A پیدا میشود [۴].

برای نقاط با ابعاد بالاتر، نمودار دیدنگار دچار پیچیدگی هایی می شود. در محیط سه بعدی، دیدنگار می تواند از رئوس چند وجهی ساخته شود، اما کوتاه ترین مسیر دیگر بر نمودار واقع نخواهد بود. لذا این نمودار بیشتر برای محیط های دو بعدی استفاده می شود.

۲-۱-۲ نمودار ورونویی[8]

نمودار ورونویی یا *محور میانی*[9] عبارت است از مکان هندسی نقاطی که در فاصله یکسانی از اجسام ریاضی (مثلاً نقطه یا خط) قرار دارند. این نمودار فضا را به نواحی ای تقسیم می کند که هر ناحیه یک خصوصیت دارد. برای هر نقطه در ناحیه، این خاصیت، نزدیک ترین به نقطه بودن است. یک مسیر بین نقطه شروع و هدف با حرکت از موقعیت شروع و رسیدن به هدف روی نمودار، به دست می آید. شکل ۱-(ب) نمودار ورونویی ساده (اجسام نقطه اند) و ۱-(ج) نمودار ورونویی تعمیم یافته (اجسام خط اند) را نشان می دهد.



الف   ب   ج

شکل ۱ – (الف) دیدنگار؛ (ب) نمودار ورونویی ساده برای ۱۱ نقطه؛ (ج) نمودار ورونویی تعمیم یافته.

نمودار ورونویی از دو جنبه جالب توجه است: تنها $O(n)$ لبه در نمودار وجود دارند و می توانند به طور مؤثری در زمان $O(n\log n)$ ساخته شود، که n تعداد نقاط است. مزیت دیگر این است که همبند بودن فضای اولیه مستقیماً به نمودار منتقل میشود، در حالی که در سایر روش ها باید همبند بودن را به طور مصنوعی در گام پردازش ثانویه ایجاد شود [۵]. برای ابعاد بالاتر از دو بعدی، نمودار ورونویی ناکارآمد بوده و ساختارهای دادهٔ پیچیده ای نیاز دارد. علی رغم این موضوع، نمودار ورونویی در مسائل برنامه ریزی روبات مخصوصا در ابعاد پائین کاربرد های بسیار وسیعی دارد.

۲-۱-۳ شبکه زیر هدف[10]

اساس کار در این روش تولید موقعیت هایی میانی است که در هر حرکت به عنوان زیر هدفِ هدف اصلی محسوب می شوند و یک عملگر وظیفه ذخیره سازی این موقعیتها را بر عهده دارد. اگر مجموعه این زیر هدف ها در نهایت به هدف اصلی منتهی نشد، آنها را در مرحله بعد به حساب نمی آورد. مهمترین مزیت این الگوریتم این است که به حافظه کمی احتیاج دارد هر چند که نمایش دقیقی از اشکال موانع میسر نخواهد بود.

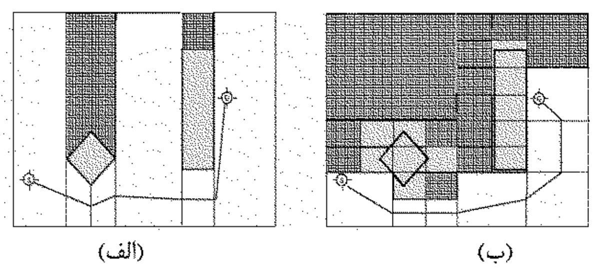
۲-۱-۴ طرح پیرامونی[۱۱]

یک راه عمومی برای ساخت مسیر در ابعاد اختیاری در [۶] ارائه شده است. این روش طرح یک جسم را از ابعاد بالاتر به ابعاد پایین تر می کشد و سپس منحنی های مرز جسم تصویر شده را دنبال می کند. این کار تا زمانی ادامه می یابد که خطوط، یک بعدی شوند و مسیر نهایی روی همین نمودارهای یک بعدی پیدا می شود که پیچیدگی کمتری دارند.

۲-۱-۵ تجزیه سلولی[۱۲]

در این روش فضای خالی محیط کاری به مجموعه ای از سلولها تقسیم بندی می شود. سلولها باید ساده باشند که نقشه مسیر به راحتی در هر سلول ایجاد شود (سلولهای مناسب اغلب چند ضلعی های محدب هستند). کانالی[۱۳] از سلولهای خالی متوالی که با سلولی که شامل موقعیت فعلی روبات می باشد، شروع، و با سلولی که موقعیت نهایی روبات را دارد اتمام می یابد، همان مسیر حرکت روبات را شامل می شود. تقسیم بندی سلولی به دو دسته دقیق[۱۴] و تقریبی[۱۵] تقسیم می شود. تفاوت اصلی این دو روش این است که، روش تقریبی تقسیم بندی سلولی راحت تر انجام می شود ولی هیچ تضمینی وجود ندارد که در صورت وجود داشتن مسیر حرکت روبات، یک مسیر آزاد برای حرکت آن بیابد.

در شکل ۲-(الف) روش تقسیم بندی سلولی دقیق ذوزنقه ای را نشان می دهد و در قسمت (ب) نتیجه روش تقسیم بندی سلولی تقریبی همان محیط را نشان می دهد. بهترین درجه پیچیدگی محاسباتی این روش تاکنون $O(n\log n)$ می باشد. محدودیت روش تقسیم بندی سلولی این است که تنها در محیط های ایستا قابل استفاده هستند (جایی که موانع حرکت نمی کنند). در محیط هایی که در آنها موانع نیز حرکت می کنند و به محیط های پویا معروفند، این الگوریتم ها کارایی لازم را ندارند [۷].



(الف) (ب)

شکل ۲ – (الف) روش تقسیم بندی سلولی دقیق ذوزنقه ای؛ (ب) نتیجه روش تقسیم بندی سلولی تقریبی همان محیط.

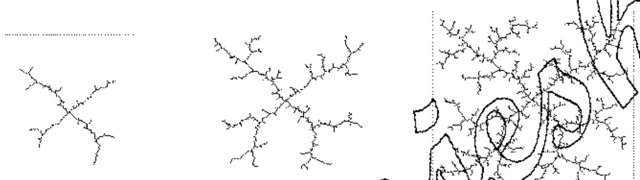
۲-۱-۶ روش نقشه مسیر احتمالی (PRM)[۱۶]

این روش جزء الگوریتم های احتمالی است که در سالهای اخیر رنج وسیعی از مسائل روباتیک را در بر گرفته اند. الگوریتم های احتمالی، هسته اصلی حوزه روبات های اتوماتیک را تشکیل می دهند و از نظر عملی و مقاوم بودن، پیشرفتهای بی سابقه ای داشته اند. در این روش ابتدا نقاطی بطور تصادفی ایجاد می شوند، سپس نمونه هایی که به روی موانع قرار گرفته اند از مجموعه نقاط موجود حذف می شوند. در مرحلهٔ بعد نقاط بدست آمده با شرط عدم برخورد با موانع به یکدیگر متصل می شوند و نقاط شروع و پایان حرکت روبات، به گراف بدست آمده افزوده می شوند. حال با استفاده از روشهای جستجوی گراف می توان مسیر حرکت روبات را بدست آورد.

یکی دیگر از نکات مهمی که باید مورد توجه قرار گیرد، کنترل تعداد نقاط ایجاد شده توسط الگوریتم است. زیرا با افزایش تعداد نقاط پردازش های اولیه مشکل تر می شود و زمان لازم جهت اجرای الگوریتم طولانی می شود. به همین دلیل الگوریتم طوری طراحی می شود که تعداد نقاط از عددی که توسط کاربر تعیین می شود، بیشتر نشود. نکته دیگری که باید مورد توجه قرارگیرد، تعداد اتصالات هر گره با گره های مجاور آن می باشد، زیرا با افزایش تعداد این اتصالات تحلیل گراف دشوارتر می شود و به همین دلیل نیز باید به مقدار خاصی محدود شود [۸].

۲-۱-۷ روش درخت های احتمالی با رشد سریع (RRT)[۱۷]

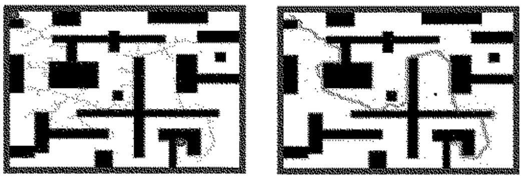
این روش هم یکی از الگوریتم های احتمالی است که در سالهای اخیر مورد استفاده زیادی قرار گرفته است. طرز عمل بدین صورت است که از یک نقطه شروع، نقاطی در همسایگی آن بطور تصادفی ایجاد شده و طوری به نقطه قبلی وصل می شوند که تشکیل گراف دور دار نکنند (یعنی یک درخت تشکیل شود) (شکل ۳). از مزایای این الگوریتم می توان به سرعت و دقت بالای آن اشاره نمود. همچنین در این روش هیچگونه گسسته سازی به روی فضای حرکت روبات انجام نمی گیرد و الگوریتم کاملا به صورت پیوسته عمل می کند. این امر باعث می شود که خطای گسسته سازی محیط به طور کامل حذف شود.



شکل ۳- نمونه ای از گسترش گراف RRT در محیط حرکت روبات.

یکی از اشکالات این روش این است که گراف حاصل در تمام جهات به طور یکسان رشد می کند، این موضوع باعث می شود که زمان لازم جهت رسیدن به هدف طولانی شود. برای رفع این مشکل از الگوریتم های اُریب[۱۸] استفاده می شود که این امر سبب می شود که نمودار در جهت هدف گسترش بیشتری داشته باشد. اما اگر مقدار اُریب زیاد باشد، باعث توقف الگوریتم در نقاط مینیمم محلی خواهد شد. برای سرعت بخشی به الگوریتم دو گراف مجزا یکی در سمت نقطهٔ شروع حرکت روبات و دیگری در سمت نقطهٔ پایان به طور همزمان شروع به گسترش می کنند و هدف الگوریتم اتصال

گراف های ایجاد شده به یکدیگر می باشد. پس از تشکیل گرافی متشکل از نقاط شروع و پایان حرکت روبات، می توان با جستجوی گراف مسیر مناسب جهت هدایت روبات را بدست آورد (شکل ۴) [۹].



شکل ۴ - چگونگی مسیر یابی روبات در یک محیط پر تراکم و پیچیده بوسیلهٔ روش RRT.

۲- ۲ روش های دیگر

۲- ۲- ۱ برنامه ریزی ریاضی

برنامه ریزی ریاضی شیوه دیگری برای یافتن راه حل بهینه است. با معرفی فاصله روبات از مانع به عنوان یک متغیر، برنامه‌ریزی حرکت به عنوان یک مسئله بهینه سازی ریاضی فرموله می‌شود که یک مسیر بین موقعیت شروع و هدف پیدا میکند در حالی که یک مقدار اسکالر قطعی را کاهش می‌دهد. این مقدار معمولاً مسافت کلی مسیر است. مدلهای ریاضی مختلفی، که تکنیک‌های آنها از الگوریتم های بهینه سازی یا محاسبات عددی و هندسه بهره می برند، توسعه داده شده اند.

۲- ۲- ۲ میدان های پتانسیل[۱۹]

در رویکرد میدانهای پتانسیل، میدان پتانسیل برای تمامی فضای کاری با استفاده از دانش کافی نسبت به موانع و نقطه هدف محاسبه می شود. همین میدان مشخص کننده مسیر حرکت روبات از نقطه شروع تا نقطه پایان می باشد. [۱۲]. یک روبات در این شیوه به عنوان یک نقطه در موقعیت فضا مطرح می شود. تابع پتانسیل می تواند در فضای آزاد به صورت جمع پتانسیل های جاذب که روبات را به سمت موقعیت هدف می رانند و پتانسیل های دافع که آن را از موانع دور می کنند، تعریف شود [۴].

۲- ۲- ۳ شبکه های عصبی

در [۱۳] از رویکرد شبکه عصبی برای برنامه ریزی حرکت بدون برخورد در زمان واقعی (به هنگام[۲۰]) برای روبات های متحرک و بازویی در محیط های پویا استفاده شده است. در [۱۴] یک شاخه جدید از مبحث شبکه عصبی با نام شبکه عصبی توسعه امواج متغیر[۲۱] معرفی شده است و از آن برای تولید مسیر حرکت روبات های متحرک و بازویی در محیط های پویا استفاده شده است. شبکه‌های هاپفیلد هم نمونه دیگری از کاربرد شبکه‌های عصبی در حل مسائل برنامه‌ریزی حرکت روبات می‌باشند.

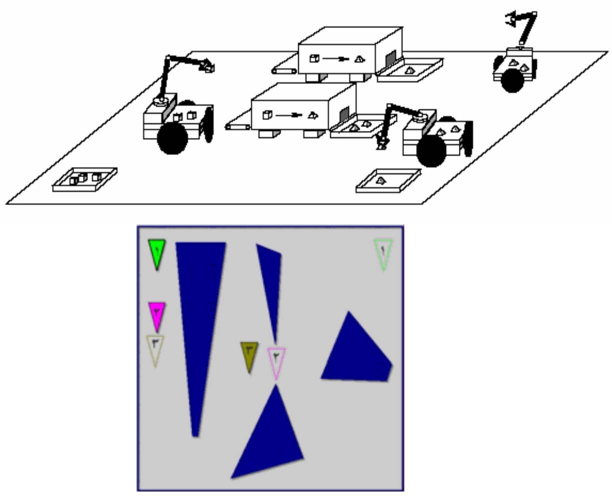
۲- ۲- ۴ الگوریتم ژنتیک

الگوریتم ژنتیک، الگوریتمی ریاضی است که با استفاده از الگوهای عملیاتی قاعده داروینی در مورد تکثیر بقای احسن و بر اساس فرآیند طبیعی ژنتیک (ترکیب جنسی)، مجموعه ای (جمعیت) از اشیای منفرد ریاضی - معمولا رشته های کاراکتری با طول ثابت به عنوان رشته کروموزومها- با میزان تطبیق خاصی را به جمعیت جدیدی (برای مثال نسل بعد) تبدیل می کند. الگوریتم ژنتیک فضای جواب را جستجو می کند تا رشتهٔ دارای بالاترین میزان تطبیق را از میان رشته های کاراکتری ممکن بیابد. در حل مساله برنامه ریزی حرکت روبات، رشته کروموزومی، توالی نقاطی است که نقطه شروع روبات را به پایان آن وصل می کنند. الگوریتم ژنتیک طی تکرار های متعدد و با تغییر و اصلاح این رشته، در تلاش برای بهینه سازی یک معیار مشخص مانند طول مسیر، کوتاه ترین مسیر را برای روبات پیدا می کند.

۳ مسأله برنامه ریزی حرکت چند روباتی[۲۲]

در بسیاری از مواقع برنامه ریزی حرکت روبات به گونه ای است که با مجموعه ای از روبات ها مواجه هستیم و موقعیت کاری روبات بسیار پیچیده تر خواهد بود، زمانیکه روبات مجبور باشد فضای کاری خود را با روبات های دیگر به اشتراک گذارد. نمونه ای از کاربردهای این دسته از روبات ها در سیستمهای حمل و نقل مدرنی است که در فرودگاه ها، گمرک و کارخانجات دیده می شود. مساله ای که در برنامه ریزی حرکت سیستمهای چند روباتی با آن روبرو هستیم از قرار زیر است:

مجموعه ای از روبات ها $R_1, R_2, \ldots, R_n$ با درجات آزادی به ترتیب $d_1, d_2, \ldots, d_n$ موجود است که در فضای کاری $n \times W$ حرکت می کنند. هدف مسئله یافتن مسیری است که هر روبات را به طور مجزا از موقعیت موجودش به نقطهٔ هدف آن برساند، به نحوی که هیچ نوع برخوردی با موانع و روبات های دیگر نداشته باشد یا اینکه گزارش دهد که مسئلهٔ مورد بررسی جواب ندارد (شکل ۵) [۱۱].



شکل ۵- حرکت سه روبات از یک آرایش اولیه به آرایش نهایی باید طوری انجام شود که هیچ برخوردی صورت نگیرد.

بسیاری از مزایای روبات ها فقط زمانیکه از ترکیب آنها استفاده می کنیم حاصل می شوند، که به برخی از آنها در ذیل اشاره شده است [۱۱].

- توانایی کاری: راه حل بسیاری از مسائل به گونه ایست که بصورت موازی باید حل شوند و احتیاج به چندین روبات دارند.
- بهبود اجرای سیستم: حل در زمان کوتاهتر و رسیدن به کارایی بیشتر نیازمند ترکیب روبات ها است.
- مشاهدات توزیع شده: روبات های سنسور دار به داده های مشاهده شده از موقعیت های مختلف نیاز دارند تا بتوانند همزمان به نتیجه گیری درستی دست یابند و این کار از عهده یک روبات به تنهایی بر نمی آید.
- تلرانس خطا: اگر تحت شرایطی یک روبات دچار مشکل شود، حیات کل سیستم به مخاطره نمی افتد.

با اینهمه، ترکیب چندین روبات برای حل یک مسئله فقط دارای مزیت نیست بلکه ما احتیاج داریم که ملاحظات طراحی و برنامه ریزی خاصی را مد نظر قرار دهیم، مثلاً

۱. وظایف چگونه باید بین روبات ها تقسیم شوند؟
۲. چگونه باید منابع موجود را بین روبات ها تقسیم کرد؟

پاسخ به سوال اول بستگی به نوع تقسیم بندی مجموعه روبات های مورد استفاده دارد. اگر این مجموعه از روبات ها همگن[۲۳] باشند، این بدان معنی است که روبات ها دارای تواناییهای یکسانی هستند. اگر در حالت دیگر روبات ها دارای توانایی های مختلفی باشند آنها را غیر همگن[۲۴] می‌نامند. توزیع وظایف در حالتی که با این دسته از روبات ها مواجه هستیم بسیار پیچیده‌تر است، مخصوصا زمانی که وظیفه‌ای خاص فقط توسط یک نوع خاص از روبات ها انجام می گیرد.

در مورد سوال دوم، منابع همان ابزارهایی هستند که روبات ها برای تحقق وظایف خود به آنها احتیاج دارند. به عنوان مثال فضای کاری. در عمل، تقسیم بندی فضا بدان معنی است که از برخورد بین روبات ها باید جلوگیری شود. با تقسیم بندی منابعی از این قبیل مشکلی به نام بن بست یا وقفه[۲۵] هم ایجاد می شود. روباتی که مؤدب[۲۶] قلمداد می شود منابع خود را برای دیگران از دست می دهد، و ممکن است مجبور شود که برای همیشه منتظر بماند. با هماهنگی بین روبات ها در سیستم های چند روباتی امید آن می رود که این مشکل حل شود [۱۲].

هدف اصلی در برنامه ریزی حرکت چند روباتی آن است که هم هیچ برخوردی بین روبات ها با هم و با موانع در طول مسیر حرکتشان رخ ندهد و هم رخداد بن بست یا وقفه اتفاق نیفتد. دو رویکرد در رابطه با برنامه ریزی حرکت سیستمهای چند روباتی پیشنهاد شده است. یک رویکرد این است که همهٔ روبات ها به صورت یک روبات واحد در فضای پیکربندی (مجموعه ای از روبات ها با تعداد زیادی درجات آزادی) در نظر گرفته می شود و لازم است که هم پیکربندی اولیه و هم پیکربندی نهایی کل روبات ها مشخص باشد. بدین معنی که برنامه ریزی حرکت روبات ها در یک فضای پیکربندی شدهٔ مرکب صورت می گیرد و این روش را اصطلاحاً *برنامه ریزی متمرکز*[۲۷] می نامند. عدم برخورد یک روبات با روبات های دیگر در این سیستم به معنای عدم برخورد قسمتی از یک روبات واحد با سایر قسمتهای همان روبات است. از مزایای این روش کامل بودن آن است و بزرگترین عیب آن اینست که بعد فضای پیکربندی شده روباتِ واحد بسیار بزرگتر از بُعد هر روبات آن به صورت مجزا است. این امر باعث افزایش پیچیدگی محاسباتی و زمانی مسئله می شود.

رویکرد دیگر که روش بسیار متداول تری است *برنامه ریزی تفکیکی*[۲۸] نامیده می شود. در این روش برنامه ریزی برای هر روبات طی دو مرحله انجام می گیرد: در مرحلهٔ اول، برنامه ریزی ابتدایی برای هر روبات به طور جداگانه انجام می گیرد و در مرحلهٔ دوم این برنامه با در نظر گرفتن موقعیت سایر روبات ها تصحیح می شود. الگوریتم هایی که این رویه را دنبال می کند اغلب کامل نیستند، بدین معنی که هیچ نوع تضمینی نیست که اگر راه حلی وجود داشته باشد حتماً آن را پیدا کنند.

۴ برنامه ریزی حرکت چند روباتی با استفاده از گراف

نظریه گراف، برخلاف بسیاری از سایر روشها، این ویژگی را دارد که می تواند حرکت چند روبات را حتی با حرکات همزمان برنامه ریزی نماید. به علاوه، نظریه گراف به ویژه در حوزه برنامه ریزی حرکت چند روباتی قابلیت هایی دارد که در ادامه تشریح خواهند شد.

به منظور مدل کردن و بیان مساله برنامه ریزی حرکت چند روباتی به صورت یک مساله گراف، چند پارامتر اساسی باید به طور مشخص در نظر گرفته شوند. با توجه به تعریف مساله به صورت طراحی یک مسیر بهینه قابل حل و ارائه قوانین حرکت، این

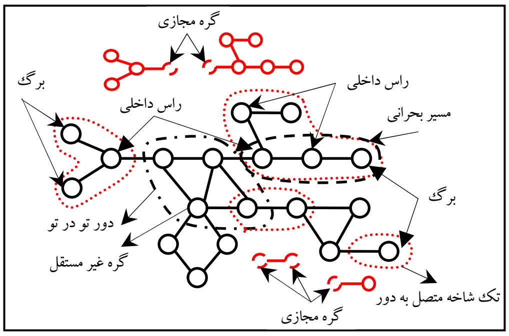
پارامترها عبارتند از: توپولوژی گـراف، پیکربنـدی (آرایـش) اولیـه و پیکربندی نهایی روبات ها روی گراف، قابل حل بودن کلـی و جزئـی گراف برای تعداد روبات مشخص، بهینه بودن گراف، و قوانین حرکت روبات ها روی گراف. با داشتن برخی از این پارامترها به عنوان فرض و به دست آوردن برخی دیگر به عنوان حکم، مـی تـوان مـسائلی را تعریف کرد که پاسخ مساله برنامه ریزی حرکت در پاسخ این مسائل است. منظور از قابل حل بودن و بهینه بودن گراف بشرح زیر است:

- گراف قابل حل کلی: چنانچه رسیدن از هر آرایش اولیه روبات هـا روی گراف به هر آرایش نهایی از طریق حرکات متوالی روبـات هـا بر روی یال های گراف، بدون برخورد یکدیگر امکـان پـذیر باشـد، گراف قابل حل کلی است.
- گراف قابل حل جزئی: چنانچه رسیدن از تنها برخی آرایـش هـای اولیه روبات ها روی گراف به برخی آرایـش هـای نهـایی از طریـق حرکات متوالی روبات ها بر روی یال هـای گـراف، بـدون برخـورد یکدیگر امکان پذیر باشد، گراف قابل حل جزئی است.
- گراف قابل حل مینیمال: چنانچه تعداد رئـوس (گـره هـای) یـک گراف قابل حل کلی به حدی باشد که حذف یک رأس قابلیت حل کلی گراف را از بین ببرد، آن گراف قابل حل مینیمال است.
- گراف بهینه: بهینه بودن گراف بر اساس زمان رسیدن از پیکربندی اولیه به پیکربندی نهایی، یا تعداد رأس هـای گـراف، و یـا هـر دو تعریف می شود. هرچه تعداد رأس ها و زمـان حـل مـسأله کمتـر باشد گراف مطلوب تر است. از طرفی زمـان حـل بـه ایـن عوامـل بستگی دارد: تعداد حرکات روبات ها، سرعت حرکـت روبـات هـا، امکان یا عدم امکان حرکت تـوأم روبـات هـا، و تعـداد رأس هـای گراف. کم بودن رأس ها ممکن است باعث شود تعداد حرکت های زیادی برای رسیدن به پیکربندی نهایی لازم باشد و این یعنی بالا رفتن زمان. بنابر این حالت بهینه حالتی اسـت کـه تعـادلی میـان تعداد رأس ها و زمان حل حاصل شود. در شرایطی ممکـن اسـت تنها کم بودن تعداد رأس ها مهم باشد و زمان یـا تعـداد حرکـات روبات ها مهم نباشد. در این حالت گراف بهینـه، گـراف مینیمـال است. همچنین ممکن اسـت فقـط زمـان بـرای مـا مهـم باشـد و محدودیتی در افزایش تعداد گره های گراف مسیر نداشته باشیم.

علاوه بر چند تعریف رایج در نظریه گراف، تعاریف دیگری برای تشخیص نوع گراف (قابل حـل کلـی یـا جزئـی، بهینـه و مینیمـال) توسعه داده ایم که در جدول ۱ آورده شده اند. در شکل ۶ درخت ها همراه با گره های مجازی به طور جداگانه رسم شده اند.

جدول ۱- چند تعریف مرتبط با گراف های قابل حل.

اصطلاح	شرح
حرکت	جابجایی یک روبات از یک گره به گره مجاور از طریق یال متصل کننده آن دو به هم
پیکربندی اولیه	چیدمان اولیه روبات ها روی گره ها
پیکربندی نهایی	چیدمان نهایی روبات ها روی گره ها
درخت (T)	بخش هایی از گراف که هیچ دوری ندارد. هر درخت یک بخش از گراف را تشکیل می دهد.
برگ (L)	هر گره از درخت با درجه ۱
گره دور	گره هایی که متعلق به دور هستند.
رأس داخلی(B)	هر گره از درخت با درجه بزرگتر یا مساوی ۱
گره مستقل	در یک دور گره ای مستقل است که با گره دور دیگری مشترک نباشد.
گره غیر مستقل	اگر یک گره بین دو دور مشترک باشد گره غیر مستقل است.
تک شاخه متصل به دور	یک یال و یک گره که به صورت برگ به یکی از گره های دور متصل شده اند
|G|	تعداد کل گره های گراف
مسیر بحرانی	بعد از جدا کردن درخت ها و برگ ها و تعیین گره های مجازی، طولانی ترین مسیر از بین مسیر هایی که از گره مجازی شروع می شود یا بین دو گره مجازی قرار دارد و از هیچ گره دوری عبور نمی کند
B*	تعداد گره های مسیر بحرانی
سَر	گره ابتدایی هر مسیر
دُم	گره انتهایی هر مسیر
حرکت بین بخشی	حرکت روبات ها از یک بخش به بخش دیگر
گراف های i دوری	اگر گراف i دور مجزا داشته باشد به آن i دوری می گویند
دور تو در تو	دور است که در داخل دور دیگر وجود دارد.



شکل ۶- تعاریف اجزای گراف.

از آنجایی که گرافها با نام گذاری رأس ها و یال ها و سایر اجـزا و ویژگی هایشان، توانائی خوبی در کمـک بـه بیـان قـوانین حرکـت روبات ها و اولویت حرکت آنها دارند، به کمک ایـن نظریـه خـواهیم توانست گراف مینیمال قابل حل را برای مسأله برنامه ریزی حرکـت خود به دست آوریم.

از آنجا که پاسخ مساله برنامه ریزی حرکت در پاسخ به مسائلی است که از جابجائی پارامترهای اساسی مساله برنامه ریـزی حرکـت به عنوان فرض یا حکم ایجاد می شوند، می توان گفت مساله برنامـه ریزی حرکت شامل مجموعه ای از زیر مـساله هاسـت. از بـین ایـن مسائل با توجه به چارچوب تعیین شده می تـوان مـساله زیـر را بـه عنوان مسأله جامع مورد نظر، تعریف کرد:

با فرض اینکه m روبات روی یـک گـراف معـین دارای آرایـش اولیه خاصی می باشند و می خواهند به آرایش نهایی خاصی دسـت یابند. آنگاه:

- آیا این گراف برای این تعداد روبات در حالت کلی قابل حل است؟
- آیا این گراف برای این تعداد روبات و رسیدن از این آرایش به آرایش نهایی مورد نظر قابل حل است؟
- آیا این گراف برای حل این مسأله گراف مینیمال است؟
- آیا این گراف برای حل این مسأله گراف بهینه است؟
- قوانین حرکت روبات ها برای رسیدن به آرایش نهایی چگونه باشد تا در کوتاهترین زمان به این هدف برسیم؟

ملاحظه می شود که تعداد روبات ها، توپولوژی گراف، پیکربندی اولیه و پیکربندی نهایی فرضهای مسأله و پاسخ سؤالات فوق حکم های مسأله هستند. شرایط مساله به لحاظ ویژگی های چهار عامل یعنی گراف، محیط، روبات ها و موانع نیز باید برای حالت های مختلف مشخص شوند.

در ادامه مساله هایی را تعریف می کنیم که جزئی از این مساله جامع هستند. هر مساله برای نشان دادن یکی از برتری های نظریه گراف می باشد. فرض ها، حکم ها و شرایط مساله به لحاظ ویژگی های عواملی مانند گراف، محیط، روبات ها و موانع در هر مساله مشخص می شوند. فرضیات و شرایط مساله عبارتند از:

- تعداد روبات ها برابر است با m.
- توپولوژی گراف یک گراف کلی است و می تواند دور داشته باشد.
- گراف مسطح، همبند، بدون جهت و ساده می باشد.
- تنها گره ها می توانند محل توقف روبات ها باشند
- روی هر گره و هر یال حداکثر یک روبات قرار می گیرد.
- سرعت حرکت تمام روبات ها روی یال ها یکسان است.
- برای سادگی مسأله فرض می شود طول همه یال ها یکسان و برابر واحد است. بنابراین زمان حرکت روبات روی هر کدام از یال ها برابر می باشد.
- روبات ها از پیش اطلاعات محیط شامل توپولوژی گراف، پیکربندی اولیه، پیکربندی نهایی، نقطه مقصد نهایی خود را دارند.
- روبات ها می توانند از حرکت روبات های دیگر با خبر شوند.
- حرکت روبات ها روی دورها می تواند به صورت همزمان صورت گیرد ولی در دیگر نواحی (درخت ها) حرکت به صورت غیر همزمان است.

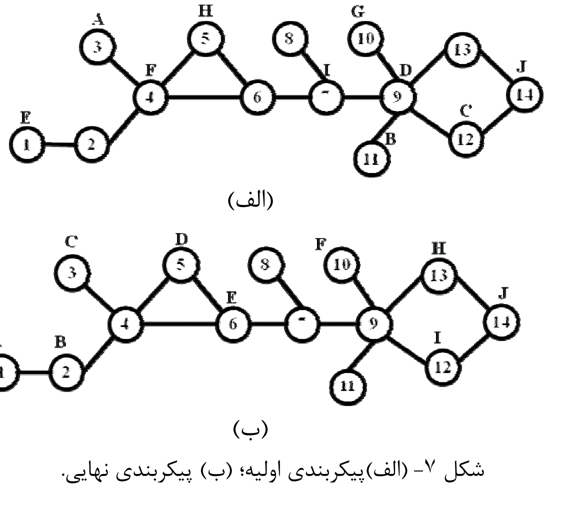
۴-۱ تشخیص گراف های قابل حل کلی

مساله تشخیص قابل حل بودن کلی یک گراف برای m روبات به دو زیر مساله تقسیم می شود: (۱) تشخیص اینکه چه گرافهایی برای m روبات قابل حل کلی هستند؟ (۲) حداکثر تعداد روبات برای اینکه گراف مورد نظر قابل حل باشد چقدر است؟

برای پاسخ به پرسش (۱) بایستی ثابت نمود که برای m روبات چه گرافهایی می توانند قابل حل باشند. مسلما اگر یک گراف با تعداد n رأس برای m روبات قابل حل باشد، تمام گراف ها با تعداد رئوس بزرگتر از n نیز برای این تعداد روبات قابل حل خواهند شد. از طرفی ممکن است با تغییر توپولوژی گراف به دست آمده با n رأس به گرافی دست پیدا کنیم که تعداد رأس هایش کمتر از n باشد و برای m روبات قابل حل شود. بنابراین بایستی بتوان بطور دقیق مشخص نمود که تعداد رئوس و توپولوژی گراف چه باید باشد.

قضیه ۱. جابجایی روبات بین بخش ها از طریق دورها و با کمک برگ ها صورت می گیرد.

اثبات: اگر انتقال یک روبات از یک گره به گره دیگر به سادگی امکان پذیر نباشد باید بتوان مسیر رسیدن روبات از آن گره به گره مقصد را خالی کرد. برای این کار لازم است روبات ها بتوانند بین بخش ها جابجا شوند. از آنجا که بین هر دو بخش حداقل یک دور وجود دارد برای انتقال یک روبات از یک بخش به بخش دیگر (مثلاً در شکل ۷، روبات C از گره ۱۲ به گره ۳) لازم است آن روبات را وارد دوری کنیم که بین آن دو بخش قرار گرفته است. برای این کار باید گره هایی که در مسیر رسیدن روبات C به دور هستند خالی باشند. این روبات ها روی رأس های داخلی واقع اند. بنابراین ابتدا باید روبات ها را از روی این رأس های داخلی به برگ ها منتقل کنیم. در این حالت ممکن است باز هم مسیر کاملا خالی نشود. پس لازم است روبات های باقیمانده روی رأس های داخلی مسیر وارد بخش دیگری شوند. این کار از طریق دور انجام می شود. پس از این جابجایی ها روبات می تواند وارد دور شود. وقتی روبات C به دور رسید باید مقصد و مسیر رسیدن آن به مقصد خالی باشد. باز هم با انتقال روبات ها از رأس های داخلی به برگ ها مسیر را خالی می کنیم. اگر باز هم روباتی در مسیر وجود داشته باشد از آنجا که مسیر رسیدن روبات به دور خالی شده است، روبات هایی که در مسیر رسیدن روبات C به هدف قرار دارند با وارد شدن به دور می توانند به بخش های دیگر منتقل شوند و با خالی شدن مسیر رسیدن روبات به هدف، روبات B می تواند به هدف خود دست یابد. □



(الف)

(ب)

شکل ۷- (الف)پیکربندی اولیه؛ (ب) پیکربندی نهایی.

بنابراین دورها نقش بسیار مهمی در قابل حل بودن یک گراف بازی می کنند. علاوه بر این روبات ها با قرار گرفتن روی برگ ها و خالی کردن راس های داخلی مسیر را برای حرکت خالی می کنند.

قضیه ۲. اگر روباتی نتواند به همه ی گره های گراف دسترسی داشته باشد آنگاه روباتی که بین دو دور قرار گرفته نمی تواند وارد نزدیک ترین گره هر دو دور شود، و *دیگر روبات ها نیز نمی توانند به نزدیک ترین دور خود برسند.*

اثبات: اگر روبات نتواند به همه ی گره های گراف دسترسی داشته باشد آنگاه نمی تواند به گره های بخش دیگر نیز دسترسی داشته باشد. طبق قضیه ۱ ثابت شد که جابجایی روبات بین بخش ها از طریق دورها و با کمک برگ ها صورت می گیرد. بنابراین روباتی که نتوانسته وارد بخش دیگری شود یعنی نتوانسته است وارد دوری شود و چون وارد شدن به نزدیک ترین دور ممکن تر است بنابراین نتوانسته است به نزدیک ترین دور خود وارد شود، و هم چنین روباتی که بین دو دور قرار گرفته نتوانسته وارد حداقل یکی از دورها شود که از آن طریق به بخش دیگر دسترسی داشته باشد. □

برای یک گراف مشخص طی قضیه ای حداکثر تعداد روبات هایی که روی آن گراف می توانند از محل های اولیه خود به مقصد نهایی دست یابند محاسبه می شود. بنابراین مشخص می شود که یک گراف مشخص برای تعداد مشخصی روبات به لحاظ قابل حل بودن چه وضعیتی دارد. اگر این تعداد روبات کمتر از حداکثر به دست آمده از این قضیه برای این گراف باشد، گراف موجود برای این تعداد روبات قابل حل خواهد بود.

قضیه ۳. حداکثر تعداد روباتی که گراف برای آن تعداد قابل حل کلی است برابر است با $m = |G| - B^* - 1$

اثبات: اگر هر روبات بتواند به نزدیک ترین دور خود برسد و هم چنین روباتی که بین دو دور قرار دارد به هر دو دور برسد، حرکت او به هر جای گراف امکان پذیر خواهد بود. حال به جای در نظر گرفتن همه ی روبات ها، اگر روباتی که بیشترین فاصله با نزدیک ترین دور خود را دارد بتواند وارد دور شود مسلماً روبات های دیگر نیز می توانند وارد دور شوند. با توجه به تعریف مسیر بحرانی متوجه می شویم که روباتی که در دُم مسیر بحرانی قرار گرفته است بیشترین فاصله تا نزدیک ترین دور را دارد و اگر بتواند وارد دور شود این امکان برای دیگر روبات ها نیز فراهم خواهد بود. برای اینکه روبات از دُم مسیر بحرانی به دور برسد باید مسیر رسیدن روبات به دور، یعنی رئوس داخلی مسیر و گره متصل به دور، خالی شود. بنابر این حداکثر تعداد روباتی که گراف برای آن قابل حل می باشد برابر است با تعداد کل گره ها منهای تعداد رئوس داخلی مسیر بحرانی منهای یک (که این یک برای خالی گذاشتن گره دور متصل به مسیر بحرانی است). □

۴– ۲ تشخیص گراف های قابل حل کلی مینیمال

قضیه زیر برای تعیین حداقل تعداد گره های یک گراف قابل حل کلی اثبات می شود.

قضیه ۴. با فرض m تعداد روبات ها و $G' = |G| - B^* - 1$ باشد، آنگاه:

الف) اگر $m \le G'$ بود گراف قابل حل کلی بوده و اگر $|G| = m+1$ گراف کلی حداقل است.

ب) اگر $m > G'$ گراف قابل حل جزئی یا غیر قابل حل است.

اثبات: G' حداکثر تعداد روباتی است که در یک گراف می تواند قابل حل باشد. بنابراین اگر $m \le G'$ بود گراف برای آن تعداد روبات قابل حل کلی است و اگر بیشتر بود قابل حل جزئی یا غیر قابل حل می باشد. حداقل تعداد گره های گراف در حالتی است که $B^* = 0$ باشد که در آن صورت خواهیم داشت: $\max\{m\} = |G| - 1$ ، $|G| = \max\{m\} + 1$. بنابراین حداقل تعداد گره های گراف یکی بیشتر از حداکثر تعداد روبات ها است. □

۴– ۳ تعیین قوانین حرکت روبات ها

یکی از ویژگی های گراف این است که بیان قوانین حرکت را آسان می کند. از جمله این قوانین، قانون های ابتدای بازی، قانون های حرکت و قانون های اولویت می باشد. به عنوان نمونه چند قانون در اینجا ارائه می شوند:

- در هنگام آغاز بازی، بین گروه روبات هایی که مبدأ آن رأس داخلی است و گروه روبات هایی که مبدأ آن ها برگ می باشند، و هر دو گروه به یک گره خالی اولیه دسترسی دارند و حداقل یکی از برگهای مذکور متعلق به زنجیر رأس داخلی است، این روبات ها اولویت حرکتی برابر دارند. در غیر این صورت اولویت با روباتی است که بر اساس پیکربندی اولیه بر روی رأس داخلی بوده است.
- بیشترین اولویت حرکت با روباتی است که مقصد آن به طور مستقیم قابل دسترسی است.
- در میان روبات هایی که امکان حرکت دارند، اولویت حرکت با روباتی است که مقصد آن یک رأس داخلی است و در عین حال مبدا آن مقصد روباتی دیگر است.

۵ نتیجه گیری

در صورتی که برای یک مسئله چند روباتی مسیر هایی مشخص شوند، قبل از هر چیز باید بتوان در مورد قابل حل بودن یا نبودن گراف مسیر اظهار نظر کرد، که در روشهای قبلی برنامه ریزی حرکت چند روباتی این توانائی وجود ندارد. در این مقاله توانایی های نظریه گراف در برنامه ریزی حرکت چند روباتی بیان شده است. به کمک رویکردی جدید مبتنی بر نظریه گراف، موضوع

تشخیص قابل حل بودن مساله برنامه ریزی چند روباتی (حتی پیش از اقدام به حل آن) بر روی گراف های قابل حل مینیمال برای اولین بار مطرح شده و قوانینی برای تعیین تعداد روبات های مناسب برای تردد بر روی گراف، و نیز حرکت روبات ها توسعه داده شده اند. همچنین نشان داده شد که روشهای غیر گرافی معمولاً برای حرکت یک روبات کارائی دارند، در حالیکه نظریه گراف می تواند حرکت چند روبات را حتی با حرکات همزمان برنامه ریزی نماید، چرا که با نام گذاری رأس ها و یال ها و سایر اجزا و ویژگی های گراف ها، توانائی خوبی در کمک به بیان قوانین حرکت روبات ها و اولویت حرکت آنها بوجود می آید.

مراجع

1. Konstantinos I. Tsianos, Ioan A. Sucan, Lydia E. Kavraki. *Sampling-Based Robot Motion Planning: Towards Realistic Applications*. Computer Science Review, 2007.
2. Guilherme Augusto Silva Pereiraa, Vijay Kumarb, Mario Fernando Montenegro Camposc. *Closed Loop Motion Planning of Cooperating Mobile Robots Using Graph Connectivity*. Robotics and Autonomous Systems, 2007.
3. Ananth Ranganathan, Sven Koenig. *Integrating Graph-Based and Cell-Based Planning*, IEEE Transactions on Robotics and Automation, 2007.
4. Latombe, J. C., *Robot Motion Planning*, USA Kluwer Academic Publishers, 1991.
5. Schirmacher, Hartmut, *Extracting Graphs from Three Dimensional Neuron Datasets*, Diploma Thesis. Friedrich-Alexander University, Erlangen-Nürnberg, 1998.
6. Lin, Ming C. And Canny, John. *A Fast Algorithm for Incremental Distance Calculation*, In Proc. IEEE International Conference on Robotics and Automation, 1991.
7. Hwang, Yong K., And Ahuja, Narendra, *Gross Motion Planning - A Survey*, ACM Computing Surveys, 1992.
8. Wilmarth, S. A., Amato, N. M. And Stiller, P. F., *MAPRM: A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space*, Technical Report. Texas A&M University, 1998
9. Bruce, J. & Veslo, M. *Real-Time Randomized Path Planning For Robot Navigation*. IEEE Proceedings on Intelligent Robotics & System, 2001.
10. Aronov, Boris, De Berg, Mark, Van Der Stappen, A. Frank, And Svestka, Petr. *Motion Planning For Multiple Robots.* Symposium on Computational Geometry, 1998.
11. Parker, L. E. *Multi-Robot Team Design For Real-World Applications*. Distributed Autonomous Robotic System 2, 1995.
12. Johansson, Ronnie, *Intelligent Motion Planning For A Multi-Robot System*, Ms Thesis, Royal Institute of Technology, Stockholm, Sweden, 2000.
13. Yang, S. X. & Meng. An *Efficient Neural Network Approach to Dynamic Robot Motion Planning*. Neural Networks, 2000.
14. Lebedev, D.V. *Neural Network Algorithms for Path Planning and Exploration in Robotics*. Bielefeld University, 2001.

[1] Soft Computing
[2] Robot motion planning
[3] Configuration space
[4] Completeness
[5] Path planning
[6] Visibility Graph
[7] Generalized Visibility Graph
[8] Voronoi diagram
[9] Medial axis
[10] Subgoal network
[11] Silhouette
[12] Cell decomposition
[13] Channel
[14] Exact
[15] Approximate
16 Probabilistic Roadmap Method
[17] Rapidly-exploring Random Trees
[18] Biased
[19] Potential fields
[20] Online
[21] Dynamic Wave Expansion Neural Network – DWENN
[22] Multi Robot Motion Planning Problem
[23] Homogeneous
[24] Heterogeneous
[25] Deadlock
[26] Polite
[27] Centralized Planning
[28] Decoupled Planning