نشریه علمی پژوهشی مهندسی فناوری اطلاعات مکانی

سال ششم، شماره سوم، پاییز ۱۳۹۷

Vol.6, No.3, Autumn 2018

۱۵-۳۰

این مقاله در دومین کنفرانس ملی مهندسی فناوری اطلاعات مکانی به‌عنوان مقاله برگزیده انتخاب شده است که پس از تکمیل، داوری مجدد و اخذ پذیرش در این شماره از نشریه به چاپ می‌رسد.

کاهش داده‌های خطوط سیر مکانی-زمانی با به‌کارگیری یک الگوریتم فشرده‌سازی برخط

افسانه نصیری دهج[۱]، ساناز عظیمی[۱]، رحیم علی عباسپور[۲*]

۱- کارشناس ارشد GIS، دانشکده مهندسی نقشه‌برداری و اطلاعات مکانی، پردیس دانشکده‌های فنی، دانشگاه تهران

۲- عضو هیئت علمی دانشکده مهندسی نقشه‌برداری و اطلاعات مکانی- پردیس دانشکده‌های فنی، دانشگاه تهران

تاریخ دریافت مقاله: ۱۳۹۶/۰۸/۰۸ تاریخ پذیرش مقاله: ۱۳۹۶/۱۱/۱۱

چکیده

با توسعه روزافزون دستگاه‌های همراه مجهز به سیستم تعیین موقعیت جهانی مانند گوشی‌های هوشمند همراه، حجم زیادی از اطلاعات مکانی تولید می‌شود. این داده‌ها که بیشتر به صورت دنباله‌ای از نقاط مکانی در طول زمان ذخیره و مدل‌سازی می‌شوند، خط سیر نام دارند. حجم بالای داده‌های خطوط سیر هزینه انتقال، ذخیره‌سازی و پردازش این داده‌ها را بالا برده است. برای برطرف نمودن این مشکلات، تعدادی از الگوریتم‌های فشرده‌سازی، با رویکرد کاهش تعداد نقاط مسیر مطرح شده است. در این مقاله، هفت الگوریتم نمونه‌برداری یکنواخت، داگلاس پوکر، الگوریتم نسبت زمانی بالا- پایین، پنجره متحرک، پنجره متحرک-نسبت زمانی، الگوریتم سرعت مبنا بالا-پایین و SQUISH-E (Spatial QUalIty Simplification Heuristic - Extended) مورد بحث قرار گرفتند و مزایا و معایب هر یک بررسی شد. در این میان الگوریتمSQUISH-E ، قادر به برقراری تعادل بین نسبت فشرده‌سازی و خطای فاصله اقلیدسی همزمانی است در حالی که نرخ فشرده‌سازی بالایی نسبت به سایر روش‌ها دارد. به منظور رفع این مشکل، در این مقاله راهکاری برای متغیر کردن پنجره اولویت الگوریتم SQUISH-E ارائه شد که موجب بهبود نرخ فشرده‌سازی الگوریتم می‌شود. به منظور بررسی عملکرد روش پیشنهادی، تمامی الگوریتم‌ها روی شش زیر مسیر با پیچیدگی‌های مختلف پیاده‌سازی شده و با یکدیگر از نظر معیارهایی مانند نرخ فشرده‌سازی، زمان اجرا و خطای فاصله اقلیدسی همزمانی مقایسه شدند. نتایج به‌دست آمده حاکی از بهبود عملکرد الگوریتم پیشنهادی در نرخ فشرده‌سازی، زمان اجرایی و خطای فاصله اقلیدسی همزمانی می‌باشد. زمان الگوریتم پیشنهادی نسبت به الگوریتم SQUISH-E حدود ۱۳۰ میلی‌ثانیه کاهش و نرخ فشرده‌سازی آن ۰/۰۱۵ افزایش یافته است.

کلید واژه‌ها: خط سیر مکانی-زمانی، فشرده‌سازی، الگوریتم SQUISH-E، پنجره اولویت

* نویسنده مکاتبه کننده: تهران- خیابان کارگر شمالی، بعد از تقاطع جلال آل احمد، پردیس دانشکده‌های فنی دانشگاه تهران، دانشکده مهندسی نقشه‌برداری و اطلاعات مکانی

تلفن: ۸۸۰۰۸۸۴۱

Email : Abaspour@ut.ac.ir

[DOI: 10.29252/jgit.6.3.23]

۱- مقدمه

در سال‌های اخیر، با توسعه سریع و استفاده گسترده از دستگاه‌های سیستم تعیین موقعیت جهانی (GPS)[1]، سنسورهای RFID، ماهواره‌ها و فناوری‌های ارتباطات بی‌سیم، امکان ردیابی انواع مختلف اشیای متحرک در سراسر جهان و جمع‌آوری داده‌های موقعیت تولید شده توسط اشیای متحرک افزایش چشمگیری داشته است. این ابزارها داده‌های خط سیر مکانی-زمانی را به‌صورت دنباله‌ای از موقعیت، ویژگی[2] و زمان که سه ویژگی اساسی داده‌های جغرافیایی و پایگاه داده GIS می‌باشد، ذخیره می‌کنند [۱].

با گذشت زمان و افزایش چشمگیر حجم داده‌های مسیر، مشکلاتی در ذخیره‌سازی، انتقال، پردازش و پرس‌وجو[3] از این داده‌ها ایجاد شد. از مشکلات اساسی موجود در کاربردهایی که از داده خط سیر استفاده می‌کنند، می‌توان افزایش حجم داده برای کاربران نهایی و افزونگی داده، اشغال حافظه بالا و افزایش مدت زمان دانلود و آپلود برای ارائه‌کنندگان خدمات تلفن‌های همراه[4] را نام برد [۲]. بنابراین ارائه راهکاری مناسب برای مدیریت موثر این حجم عظیم داده‌ها ضروری است.

برای مقابله با این مشکلات، دو گروه از استراتژی‌های فشرده‌سازی مسیر با هدف کاهش اندازه مسیر پیشنهاد شد [۳]. گروه اول فشرده‌سازی برون‌خط[5] است، که بعد از جمع‌آوری داده‌های خط سیر به‌طور کامل، حجم داده‌ها را کاهش می‌دهد. گروه دوم فشرده‌سازی برخط[6] است، که مستقیما مسیر را به‌عنوان یک جسم متحرک فشرده می‌کند. فشرده‌سازی داده‌های خط سیر یک روش کارآمد برای کاهش فضای حافظه، بهبود کارایی انتقال، ذخیره‌سازی و پردازش بدون از دست دادن اطلاعات و یا سازماندهی مجدد داده‌ها بر اساس الگوریتم‌های خاص است [۴].

تا به امروز، تحقیقات گسترده‌ای در ارتباط با مدیریت داده‌های حرکت در پایگاه داده اشیاء متحرک و تحلیل‌های آماری مربوط به خط سیر صورت گرفته است. یکی از ساده‌ترین و سریع‌ترین روش‌ها جهت کاهش تعداد نقاط مسیر، الگوریتم فشرده‌سازی نمونه‌برداری یکنواخت[7] می‌باشد که هر i امین نقطه از مسیر را نگه می‌دارد [۵]. در سال ۱۹۶۱، بلمن یک الگوریتم جدیدی با نام الگوریتم Bellman ارائه کرد که بر اساس برنامه‌نویسی پویا [۶]، فشرده‌سازی مسیر را با برازش رشته‌ای از پاره‌خط‌ها را به یک منحنی انجام می‌دهد. این الگوریتم تضمین می‌کند که پاره‌خط‌هایی که تعداد مشخصی از نقاط انتخاب شده از منحنی را به هم متصل می‌کنند، نزدیک‌ترین به منحنی اصلی هستند؛ اما زمان پردازش این الگوریتم بسیار زیاد است. الگوریتم ساده‌سازی داگلاس-پوکر[8] [۷] توسط داگلاس و پوکر در سال ۱۹۷۳ پیشنهاد شد که این الگوریتم بعد زمانی داده‌ها را جهت فشرده‌سازی مسیر در نظر نمی‌گرفت. الگوریتم نسبت زمانی بالا-پایین (TD-TR)[9] [۸]، الگوریتم داگلاس پوکر را با در نظر گرفتن بعد زمانی داده‌های مسیر بهبود داد. در [۹] الگوریتم پنجره متحرک[10] برای فشرده‌سازی داده‌های خط سیر به‌صورت برخط پیشنهاد شد. این الگوریتم که فشرده‌سازی را با حرکت پنجره‌ای بر روی مجموعه نقاط مسیر انجام می‌دهد، زمان اجرایی بالایی دارد. در [۱۰] محققان الگوریتم STTrace را معرفی کردند که از سرعت و جهت نقاط برای حفظ یا حذف داده‌های مسیر استفاده می‌کند. الگوریتم

[1] Global Positioning System
[2] Attribute
[3] Query
[4] Mobile Service
[5] Offline
[6] Online
[7] Uniform Sampling Algorithm
[8] Dauglas-Pauker Algorithm
[9] Top-Down Time Ratio Algorithm
[10] Opening Window Algorithm

[DOI: 10.29252/jgit.6.3.23]

فشرده‌سازی SQUISH[1] در [۱۱] پیشنهاد شد که این الگوریتم از پنجره اولویت، که تعیین کننده اولویت هر نقطه در این پنجره است، جهت فشرده‌سازی داده‌های مسیر استفاده می‌کند. به‌دلیل خطای زیاد این الگوریتم در نرخ‌های فشرده‌سازی بالا، الگوریتم بهبود یافته SQUISH- E[2] [۱۲] معرفی شد. الگوریتم SQUISH-E، فشرده‌سازی را باتوجه به خطا و نرخ فشرده‌سازی کنترل می‌کند. خلاصه‌ای از پیچیدگی زمانی، دامنه کاربرد و معیار خطای الگوریتم‌های مورد بررسی در این مقاله در جدول (۱) آمده است.

در این تحقیق روند فشرده‌سازی نقاط خط سیر در الگوریتم‌های فشرده‌سازی هندسی نمونه‌برداری یکنواخت، داگلاس پوکر، TD-TR، پنجره متحرک، پنجره متحرک-نسبت زمانی (OPW-TR)[3] ، الگوریتم سرعت مبنا بالا-پایین (TD-SB)[4] و SQUISH-E مورد بررسی قرار گرفت و مزایا و معایب هر یک از الگوریتم‌ها بیان شد. نتایج این بررسی بیانگر نرخ و زمان فشرده‌سازی بالای الگوریتم SQUISH-E بود. در الگوریتم SQUISH-E ابتدا اندازه پنجره اولویت توسط کاربر تعیین شده و ثابت بوده ولی با برقراری شرطی، اندازه پنجره افزایش می‌یابد. در این تحقیق الگوریتم فشرده‌سازی جدیدی جهت بهبود روش SQUISH-E پیشنهاد شده است که در الگوریتم پیشنهادی اندازه پنجره اولویت با توجه به فاصله هر نقطه از خط سیر تعیین می‌شود و همچنین از دو معیار فاصله اقلیدسی همزمانی و معیار تغییر جهت برای حذف و یا حفظ نقطه استفاده کرد. هدف اساسی در این تحقیق، حذف داده‌های اضافی خط سیر با حفظ ماهیت هندسی مجموعه داده‌های خط سیر می‌باشد. در پایان عملکرد و کارایی این الگوریتم‌ها بر اساس معیارهای نرخ فشرده‌سازی، زمان فشرده‌سازی و خطای فاصله اقلیدسی همزمانی ارزیابی شد.

ساختار مقاله پیشرو به این شکل است: بخش ۲ به معرفی الگوریتم‌های نمونه‌برداری یکنواخت، داگلاس پوکر، TD-TR، پنجره متحرک، OPW-TR، TD-SB و SQUISH-E و هم‌چنین معیارهایی برای ارزیابی دقت و عملکرد الگوریتم‌های فشرده‌سازی پرداخته است. در بخش ۳ روش پیشنهادی معرفی و تشریح می‌گردد. در بخش ۴ بعد از معرفی داده‌های مورد مطالعه، الگوریتم‌های مورد استفاده در تحقیق روی این داده‌ها پیاده‌سازی شده و نتایج آن مورد ارزیابی قرار گرفته است. بخش آخر به نتیجه‌گیری مباحث پرداخته است.

۲- مبانی نظری

مطالعات مربوط به الگوریتم‌های کاهش داده خطوط سیر هندسی را به‌طور کلی می‌توان بر اساس نحوه بررسی و ورود داده‌ها در الگوریتم، به دو بخش برون‌خط و برخط طبقه‌بندی کرد. در الگوریتم‌های فشرده‌سازی برون‌خط بعد از جمع‌آوری یک مجموعه کامل از نمونه‌ها، با حذف داده‌های اضافی فشرده‌سازی مجموعه داده‌ها انجام می‌شود. الگوریتم‌های برخط با اضافه شدن هر نقطه، یک تابع احتمالی محاسبه می‌شود که حذف یا حفظ نقطه را مشخص می‌کند. از کاربردهای الگوریتم‌های برخط می‌توان به مدیریت حمل و نقل و نظارت بر ترافیک شهری اشاره کرد. این بخش بررسی جامعی از الگوریتم‌های مورد استفاده در تحقیق (بخش ۲-۱) و هم‌چنین معیارهای مناسب برای مقایسه الگوریتم‌های فشرده‌سازی را ارائه می‌دهد (بخش ۲-۲).

[1] Spatial QUalIty Simplification Heuristic
[2] Spatial QUalIty Simplification Heuristic - Extended
[3] Opening Window Time Ratio Algorithm
[4] Top-Down Speed-Based Algorithm

[DOI: 10.29252/jgit.6.3.23]

جدول۱ : خلاصه‌ای از الگوریتم‌های فشرده‌سازی مسیر (n تعداد نقاط مسیر اصلی است.)[۴]

الگوریتم	برخط/برون‌خط	پیچیدگی زمانی	معیار خطا
نمونه‌برداری یکنواخت	برخط/برون‌خط	O(n)	نرخ فشرده‌سازی
داگلاس-پوکر	برون‌خط	O(nlogn)	خطای مکانی
TD-TR	برون‌خط	O(nlogn)	خطای فاصله اقلیدسی همزمانی
OPW	برخط	$O(n^2)$	خطای مکانی
OPW-TR	برخط	O(nlogn)	خطای فاصله اقلیدسی همزمانی
TD-SP	برخط/برون‌خط	$O(n^2)$	تفاوت سرعت
SQUISH-E	برخط/برون‌خط	O(logn)	خطای فاصله اقلیدسی همزمانی، نرخ فشرده‌سازی

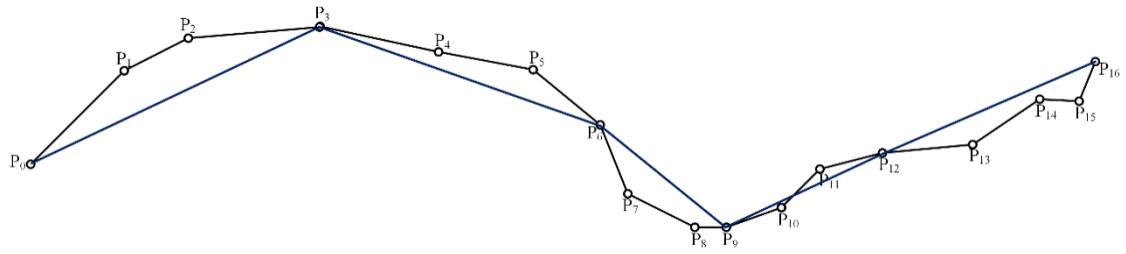
۲-۱- الگوریتم‌های فشرده‌سازی

این بخش، به معرفی الگوریتم‌های فشرده‌سازی مورد استفاده تحقیق، پرداخته است.

۲-۱-۱- الگوریتم نمونه‌برداری یکنواخت

الگوریتم نمونه‌برداری یکنواخت هر i امین نقطه (برای نمونه پنجمین، دهمین و ...) از مسیر اصلی را نگه می‌دارد [۵]. برای مثال در شکل (۱) الگوریتم نمونه‌برداری یکنواخت با حفظ هر سومین نقطه از مسیر اصلی، مسیر را فشرده می‌سازد. ایده اصلی این روش، حفظ نقاط از مسیر اصلی بدون توجه به روابط بین نقاط همسایه می‌باشد. مسیر جدیدی که توسط فرایند نمونه‌برداری یکنواخت تولید می‌شود، یک مسیر تقریبی است که جزئیاتی در مورد ویژگی‌های هندسی مسیر ندارد. از این رو برای کاربردهای خاصی که نیاز به جزئیات بیشتری از مسیر دارند، مفید نمی‌باشد. از این الگوریتم می‌توان برای کاهش تعداد نقاط مسیر اصلی و حفظ فضای ذخیره‌سازی استفاده کرد.

الگوریتم نمونه‌برداری یکنواخت یک روش کاهش داده برون‌خط و برخط بوده که سریع و ساده می‌باشد. با این حال، اغلب اوقات خطای مکانی و خطای فاصله اقلیدسی همزمانی بالایی دارد. از سوی دیگر، عدم تشخیص تغییرات سرعت و جهت در نقاط مشکل دیگر این الگوریتم می‌باشد. پیچیدگی زمانی آن به‌صورت $O(n)$ (n تعداد نقاط مسیر اصلی) است.



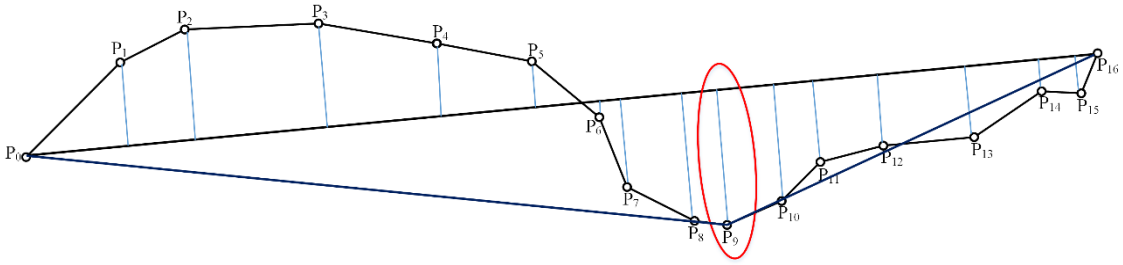
شکل ۱: الگوریتم نمونه‌برداری یکنواخت

۲-۱-۲- الگوریتم داگلاس-پوکر

الگوریتم داگلاس-پوکر یکی از معروف‌ترین روش‌های ساده‌سازی خط از دسته‌ی الگوریتم‌های برون‌خط می‌باشد. در این الگوریتم، ابتدا مجموعه کامل نقاط توسط خط واصل نقطه ابتدایی و نقطه انتهایی تقریب زده می‌شوند و خطای مکانی سایر نقاط نسبت به خط تقریب زده شده محاسبه می‌گردد. سپس این خط واصل با دو خط واصل کوتاه‌تر به‌گونه‌ای

[DOI: 10.29252/jgit.6.3.23]

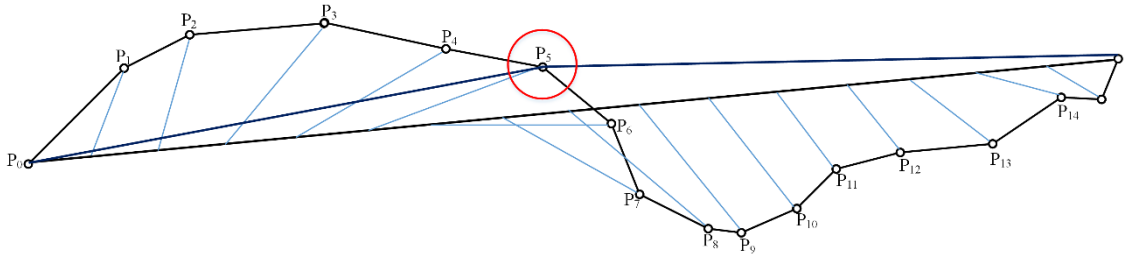
جایگزین می‌شود که نقطه انتهای خط اول و نقطه ابتدای خط دوم، همان نقطه‌ای است که بیشترین خطای مکانی را نسبت به خط واصل اولیه دارد. این فرایند تا زمانی ادامه می‌یابد که بیشترین خطای مکانی بین مسیر تقریبی و مسیر اصلی کوچک‌تر از حد آستانه موردنظر کاربر باشد. پیچیدگی زمانی این الگوریتم $O(n^2)$ می‌باشد و می‌تواند به $O(nlogn)$ بهبود پیدا کند. نقطه قوت این الگوریتم دست‌یابی به خطای مکانی کمتر از آستانه موردنظر کاربر می‌باشد و در مقابل نقطه ضعف آن استفاده از خطای مکانی بوده که موجب نادیده گرفتن بعد زمانی در فرایند کاهش داده می‌شود. برای نمونه در شکل (۲) الگوریتم داگلاس-پوکر در مرحله اول نقطه P_9 را با بیش‌ترین فاصله مکانی نسبت به خط واصل P_0P_{16} حفظ می‌کند.



شکل ۲: الگوریتم داگلاس-پوکر

۲-۱-۳- الگوریتم TD-TR

الگوریتم داگلاس-پوکر تنها ویژگی‌های مکانی نقاط خط سیر را در نظر می‌گیرد و بعد زمانی نادیده گرفته می‌شود. الگوریتم TD-TR با استفاده از فاصله اقلیدسی همزمانی به جای فاصله اقلیدسی در فرایند کاهش داده، بر محدودیت الگوریتم داگلاس پوکر غلبه می‌کند. پیچیدگی زمانی این الگوریتم $O(n^2)$ می‌باشد. الگوریتم TD-TR از تکنیک برون‌خط برای تقریب نقاط در فضای دو بعدی (مکانی-زمانی) استفاده می‌کند، تقریبا سرعت فشرده‌سازی پایینی دارد. برای نمونه شکل (۳) اولین مرحله از الگوریتم TD-TR را نشان می‌دهد که نقطه P_5 را به‌عنوان نقطه با بالاترین خطای فاصله اقلیدسی همزمانی نسبت به خط واصل P_0P_{16} شناسایی می‌کند.



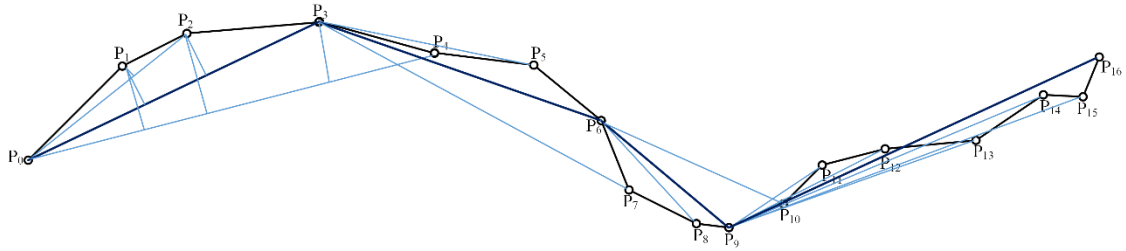
شکل ۳: الگوریتم TD-TR

۲-۱-۴- الگوریتم پنجره متحرک

الگوریتم پنجره متحرک مشابه الگوریتم داگلاس-پوکر تعدادی پاره‌خط برای تقریب خط سیر در نظر می‌گیرد. یکی از ویژگی‌های شاخص پنجره متحرک، حرکت پنجره‌ای روی نقاط خط سیر به‌منظور فشرده‌سازی خط سیر است. ابتدا این پنجره متحرک در نقطه ابتدایی قرار گرفته و به‌تدریج به سمت نقاط بعدی حرکت می‌کند تا زمانی که فاصله مکانی نقطه‌ای مانند P_j در پنجره متحرک نسبت به پاره خط واصل نقاط ابتدایی و انتهایی پنجره بزرگتر از آستانه مورد نظر (ψ) نشود. در صورت عدم برقراری شرط مذکور، نقطه ما قبل P_j به مجموعه نقاط خط سیر فشرده شده

[DOI: 10.29252/jgit.6.3.23]

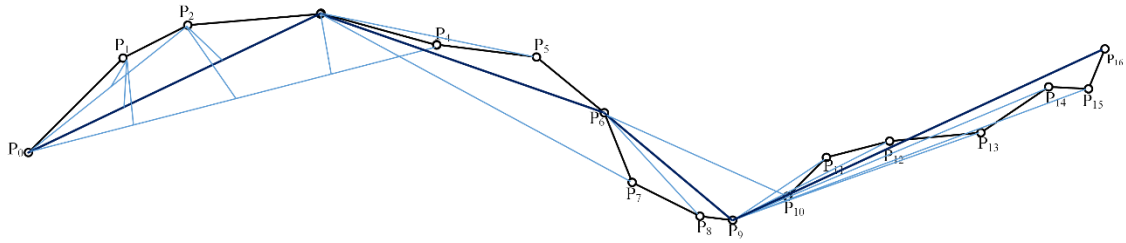
اضافه می‌گردد. نقطه اضافه شده به پنجره متحرک به‌عنوان نقطه اول برای پنجره متحرک بعدی در نظر گرفته می‌شود. این فرآیند تا زمان دستیابی به نقطه انتهایی خط سیر و فشرده‌سازی کل خط سیر اصلی تکرار می‌شود. پیچیدگی زمانی این الگوریتم $O(n^2)$ است. نقطه قوت این الگوریتم دستیابی به خطای مکانی کم‌تر از حد آستانه می‌باشد. در مقابل نقطه ضعف آن نادیده گرفتن اطلاعات ضروری خط مسیر از جمله داده‌های زمانی جهت تقریب خط سیر است [۹]. برای نمونه در شکل (۴) الگوریتم مجموعه نقاط $\{P_0, P_1, \ldots, P_{16}\}$ را با مجموعه نقاط $\{P_0, P_3, P_6, P_9, P_{16}\}$ تقریب می‌زند.



شکل ۴: الگوریتم پنجره متحرک

۲-۱-۵- الگوریتم OPW-TR

الگوریتم OPW-TR نسخه بهبود یافته الگوریتم پنجره متحرک است که از خطای فاصله اقلیدسی همزمانی به‌جای خطای مکانی استفاده می‌کند [۱۳]. این االگوریتم در مقایسه با الگوریتم پنجره متحرک، مزیت در نظر گرفتن خطای مکانی-زمانی را در تقریب خط سیر دارد. الگوریتم OPW-TR، مشابه الگوریتم پنجره متحرک دارای پیچیدگی زمانی $O(n^2)$ است. برای نمونه در شکل (۵) الگوریتم بر روی مجموعه نقاط اصلی $\{P_0, P_1, \ldots, P_{16}\}$ اعمال شد و مجموعه نقاط $\{P_0, P_3, P_6, P_9, P_{16}\}$ را از میان مجموعه نقطه اصلی حفظ کرد [۱۴].



شکل ۵: الگوریتم OPW-TD

۲-۱-۶- الگوریتم TD-SB

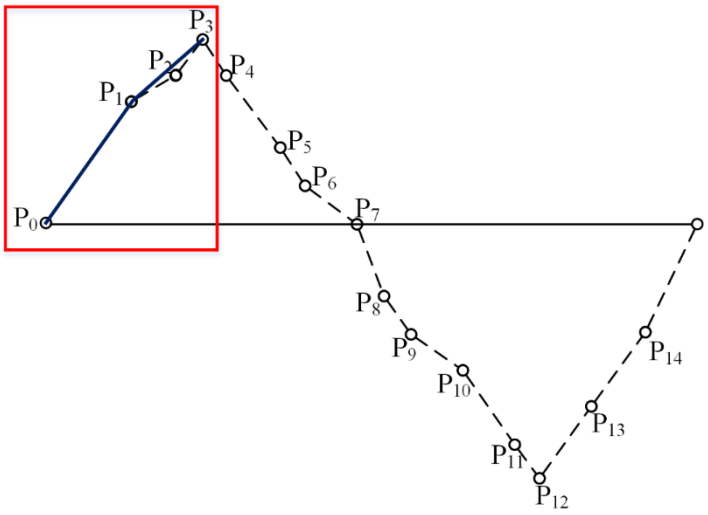
[۸] نشان داد که الگوریتم‌های فشرده‌سازی فعلی را می‌توان با بهره‌گیری از اطلاعات مکانی-زمانی موجود در سری‌های زمانی بهبود داد. الگوریتم‌های بهبود یافته از تجزیه و تحلیل سرعت‌های مشتق شده در توالی بخش‌های مسیر استفاده می‌کنند. برای نمونه در الگوریتم TD-SB، از تفاوت بین سرعت سفر دو بخش متوالی به عنوان معیاری جهت حفظ نقاط میانی استفاده شد. در این الگوریتم یک آستانه سرعت جهت حفظ نقاطی که اختلاف سرعت در آن‌ها بیشتر از آستانه مدنظر سرعت باشد در نظر گرفته می‌شود و خط سیر با نقاط دارای سرعت بالا تقریب زده می‌شود. این الگوریتم نشان داد که کاربرد مفهوم آستانه سرعت برای مسیرهای متحرک شیء مناسب است.

۲-۱-۷- الگوریتم SQUISH-E

نسخه جدیدی از SQUISH به نام الگوریتم SQUISH-E، معرفی شد که برخلاف الگوریتم‌های پیشین، خطای قابل قبولی برای فشرده‌سازی مسیر

[DOI: 10.29252/jgit.6.3.23]

ایجاد می‌کند و برای کاربران امکان فشرده‌سازی مسیر و همزمان با آن کنترل فاکتورهای نرخ فشرده‌سازی و خطای فاصله اقلیدسی همزمانی را فراهم می‌سازد. به‌عبارت دیگر، الگوریتم SQUISH-E سعی در کمینه کردن خطای فاصله اقلیدسی همزمانی μ و رسیدن به نسبت فشرده‌سازی λ را دارد. پیچیدگی زمانی الگوریتم SQUISH-E بصورت $O(n\log(n/\lambda))$ است. در این الگوریتم اولویت هر نقطه به‌صورت یک کران بالا، با درنظرگرفتن خطای SED آن نقطه که قابلیت حذف آن نقطه را مشخص می‌کند، تعریف می‌شود. در الگوریتم SQUISH-E، متغیر β ظرفیت اولویت در پنجره Q را نشان می‌دهد. برای نمونه در شکل (۶) مقدار اولیه برای پنجره اولویت برابر چهار می‌باشد. در الگوریتم SQUISH-E، مقدار ظرفیت β در صورتی که $\beta \geq i/\lambda$ باشد، افزایش می‌یابد (i تعداد نقاط بازیابی شده از مسیر T است). در این الگوریتم به هر نقطه P_i از مسیر T به‌صورت اولیه مقدار اولویت بی‌نهایت در صف Q واگذار شده است. اگر P_i اولین نقطه نباشد، P_i به‌عنوان نزدیک‌ترین جانشین از نقطه قبلی خود P_{i-1} ثبت شده و همچنین P_{i-1} به‌عنوان نزدیک‌ترین نقطه اسبق به P_i ثبت می‌شود سپس اولویت P_{i-1} با خطای فاصله اقلیدسی همزمانی سنجیده شده تا قابلیت حذف نقطه P_{i-1} را مشخص شود. وقتی پنجره Q کامل شود (شامل βتا نقطه شود)، این الگوریتم، پنجره Q را با حذف نقاطی از پنجره Q که کمترین اولویت را دارند، کاهش می‌دهد.



شکل ۶: الگوریتم SQUISH-E با پنجره اولیه برابر چهار

۲-۲- معیارها

هدف اصلی الگوریتم‌های فشرده‌سازی خط سیر، کاهش حجم داده‌ها با حفظ ماهیت هندسی خط سیر است. بنابراین، نیاز به معیارهایی مناسب جهت ارزیابی دقت و بررسی عملکرد الگوریتم‌ها است. در ادامه به بررسی نمونه‌هایی از این معیارها پرداخته می‌شود.

۲-۲-۱- خطای مکانی

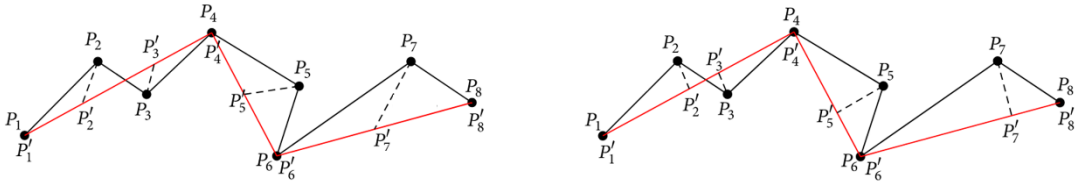
خطای مکانی عبارت است از کوتاه‌ترین فاصله بین نقطه موردنظر روی خط سیر و نقطه متناظر آن بر روی مسیر فشرده شده، که به صورت فاصله اقلیدسی عمودی نقطه از خط محاسبه می‌شود. این فاصله میزان نزدیکی نقاط خط سیر فشرده شده و نقاط خط سیر اولیه را نشان می‌دهد. خطای بین مسیر تقریبی حاصل از فرآیند کاهش داده و مسیر اصلی را می‌توان با مجموع فواصل بین نقاط برآورده شده و نقاط مسیر اولیه محاسبه کرد. این نکته حائز اهمیت است که اندازه‌گیری خطا با استفاده از مجموع و یا میانگین فواصل، نسبت به تعداد نقاط نمونه‌برداری مسیر اصلی

[DOI: 10.29252/jgit.6.3.23]

بسیار حساس است. خطای مکانی برای هر نقطه $P_i = (x_i, y_i)$ روی مسیر اصلی نسبت به موقعیت برآوردی آن $P'_i = (x'_i, y'_i)$ روی مسیر فشرده شده، با استفاده از رابطه (۱) محاسبه می‌شود.

رابطه (۱) $$(P_i, P'_i) = \sqrt{(x_i - x'_i)^2 + (y_i - y'_i)^2}$$

شکل (۷-الف) فرآیند اندازه‌گیری خطا براساس فاصله اقلیدسی عمودی بین مسیر اصلی حاصل از یک شی متحرک و یک مسیر تقریبی تولید شده با استفاده از یکی از الگوریتم‌های فشرده‌سازی مسیر را نشان می‌دهد.



الف. خطای مکانی ب. فاصله اقلیدسی همزمانی

شکل ۷: معیارهای دقت فشرده‌سازی: الف) خطای مکانی و ب) خطای فاصله اقلیدسی همزمانی

۲-۲-۲- خطای فاصله اقلیدسی همزمانی

برآورد خطای مکانی با استفاده از فاصله اقلیدسی عمودی، فقط ویژگی‌های هندسی مسیر را در نظر می‌گیرد. در سال ۲۰۰۴، مراتینا و دیبای متریک نسبت زمان- فاصله[1] را تعریف کردند. در این متریک اختلاف بین مسیر اصلی و مسیر فشرده‌شده با استفاده از تفاوت‌های جزء زمان و جزء مکان سنجیده شده است [۱۰ و ۱۴]. مزیت خطای فاصله اقلیدسی همزمانی، استفاده از داده‌های زمانی در محاسبات خطا است. در شکل (۷-ب) نمونه‌ای از الگوریتم کاهش داده‌های مسیر با استفاده از معیار فاصله اقلیدسی همزمانی آمده است.

این معیار خطا به عنوان خطای متریک جهت تقریب اعوجاج مسیر برآورد شده استفاده می‌شود. برای هر $P_i = (x_i, y_i, t_i)$ روی مسیر اصلی، موقعیت برآوردی متناظر آن روی مسیر برآوردی (x'_i, y'_i, t'_i) در صورتی‌که $t'_{ik} \le t_i \le t^i_{ik+1}$ باشد، با استفاده از روابط (۲) و (۳) محاسبه می‌گردد و در نهایت فاصله اقلیدسی همزمانی برای یک نقطه روی مسیر اصلی و نقطه متناظر همزمان روی مسیر فشرده براساس رابطه (۴) برآورد می‌شود.

رابطه (۲) $$x'_i = x'_{ik} + \frac{t_i - t'_{ik}}{t_{jk+1} - t'_{ik}} . (x'_{ik+1} - x'_{ik})$$

رابطه (۳) $$y'_i = y'_{ik} + \frac{t_i - t'_{ik}}{t_{jk+1} - t'_{ik}} . (y'_{ik+1} - y'_{ik})$$

رابطه (۴) $$ED(P, P'_i) = \sqrt{(x_i - x'_i)^2 + (y_i - y'_i)^2}$$

۲-۲-۳- زمان فشرده‌سازی

معیار دیگری که برای مقایسه عملکرد الگوریتم‌های کاهش داده خط سیر می‌توان استفاده کرد زمان فشرده‌سازی است. این معیار بیانگر مدت زمان لازم برای اجرای تکنیک فشرده‌سازی جهت کاهش داده‌های خط سیر می‌باشد [۱۵]. به عنوان مثال، زمان فشرده‌سازی ۲۴ نشان می‌دهد که زمان فشرده‌سازی مسیر اصلی ۲۴ میلی‌ثانیه است.

۲-۲-۴- نرخ فشرده‌سازی

نرخ فشرده‌سازی، نسبت اندازه مسیر فشرده شده به اندازه مسیر اصلی را مشخص می‌کند [۱۵]. رابطه (۶) نحوه محاسبه آن را نشان می‌دهد. به‌عنوان مثال نرخ فشرده‌سازی ۰٫۵ نشان می‌دهد که ۵۰ درصد نقاط در خط سیر فشرده شده باقی مانده‌اند.

رابطه (۶) $$\lambda = \frac{|T'|}{|T|}$$

[1] Time-distance ratio

[DOI: 10.29252/jgit.6.3.23]

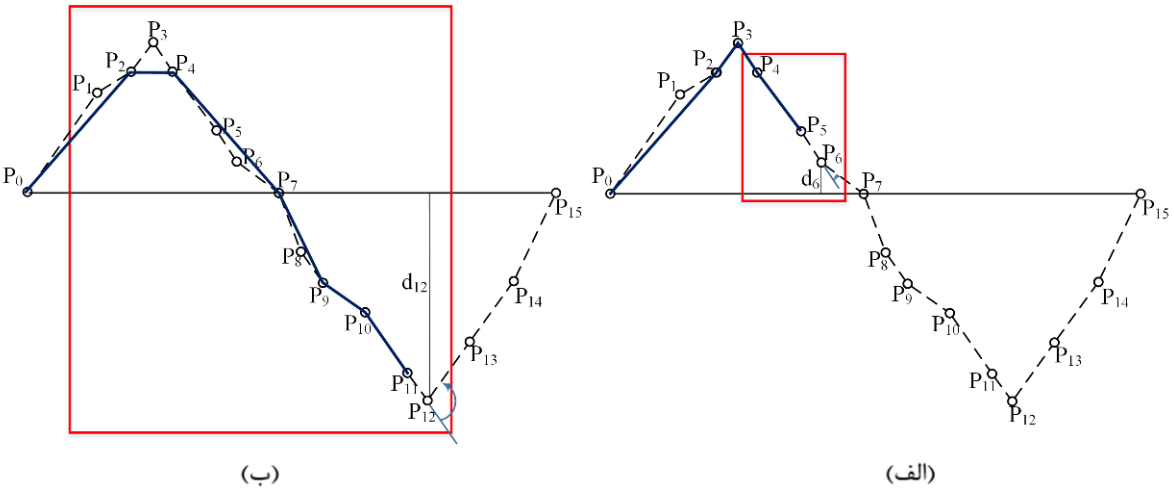
۳- روش پیشنهادی

در این تحقیق روشی برای فشرده‌سازی داده‌های خط سیر بر اساس الگوریتم‌های فشرده‌سازی مبتنی بر صف ارائه شده است که از ترکیب دو تابع فاصله و تابع جهت استفاده می‌کند. الگوریتم پیشنهادی برای فشرده‌سازی مسیر یک الگوریتم فشرده‌سازی برخط است که نیاز به تنظیم اولیه میزان پنجره اولویت دارد. این الگوریتم، خط سیر اصلی را با حذف نقاط دارای پایین‌ترین اولویت از پنجره اولویت متحرک و متغیر فشرده می‌سازد. برای این منظور بهترین زیر مجموعه‌ای از نقاط خط سیر اصلی را در پنجره اولویت در نظر می‌گیرد و با استفاده از استراتژی‌های بهینه‌سازی محلی، نقاط غیر ضروری از پنجره را تا زمانی که نقاط داخل پنجره اولویتی بالاتر از حدآستانه مورد نظر داشته باشند، حذف می‌کند.

روش پیشنهادی مشابه الگوریتم SQUISH-E می‌باشد، اما در تعیین اندازه پنجره اولویت و همچنین اولویت هر نقطه متفاوت عمل می‌کند. همانطور که در بخش (۲-۱-۷) ذکر شد، الگوریتم SQUISH-E از پنجره اولویت Q با اندازه اولیه β استفاده می‌کند و اندازه این پنجره را زمانی که شرط $i/\lambda \geq \beta$ برقرار شود، به‌صورت β + 1 افزایش می‌دهد. اندازه این پنجره توسط کاربر تعیین شده و به ویژگی‌های هندسی خط سیر وابسته نیست. در این تحقیق، الگوریتم پیشنهادی سعی در تعیین پنجره اولویت هر نقطه با توجه به معیار پیچیدگی خط دارد. معیار پیچیدگی با استفاده از فاصله عمودی نقاط خط سیر از خط واصل نقاط ابتدایی و انتهایی مشخص می‌شود. از این رو در الگوریتم پیشنهادی، ابتدا خط سیر به‌صورت خط واصل بین نقطه ابتدا و انتهای مسیر تقریب زده می‌شود و فاصله قائم هر نقطه از خط برآوردی به‌عنوان پارامتر دوری نقطه از خط سیر و یا میزان پیچیدگی خط، مطابق رابطه (۷) تعریف می‌شود.

$$d = \frac{\left|\left(\begin{bmatrix} x_1 & y_1 & 1 \\ x_n & y_n & 1 \\ x_P & y_P & 1 \end{bmatrix}\right)\right|}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2}} \qquad \text{رابطه (۷)}$$

بین فاصله هر نقطه تا خط مفروض و اندازه پنجره یک رابطه خطی به‌صورت: $F(\beta) = a.d + b$ تعریف می‌شود، بعد از تعیین ضرایب a و b، باتوجه به فاصله نقطه از خط، اندازه پنجره اولویت برای هر نقطه محاسبه می‌گردد. به عنوان مثال در شکل (۸) با توجه به فاصله دوری d_6 و d_{12} برای نقاط P_6 و P_{12} که توسط رابطه (۷) محاسبه شد، اندازه پنجره برای این دو نقطه طبق رابطه $F(\beta) = 0.75d + 2$ به ترتیب ۷ و ۳ به‌دست آمد.



شکل ۸: (الف) پنجره اولویت در روش پیشنهادی برای نقطه P_7 و (ب) پنجره اولویت در روش پیشنهادی برای نقطه P_{13} (خط سیر فشرده شده به رنگ آبی و خط سیر اصلی به رنگ سیاه می‌باشد.)

[DOI: 10.29252/jgit.6.3.23]

بعد از تعیین پنجره اولویت، اولویت هر نقطه در این پنجره بر اساس ترکیب دو خطای فاصله اقلیدسی همزمانی آن نقطه و خطای تغییر جهت خط سیر اصلی در آن نقطه نسبت به نقطه ماقبل آن، تعریف می‌شود. جهت مشخص کردن نقطه دارای کمترین اولویت، استراتژی بهینه‌سازی محلی دو هدفه به‌کار گرفته شد. روش مجموع وزن‌دار[1] یکی از روش‌های حل مسائل بهینه‌سازی دو هدفه است که در روش پیشنهادی جهت یافتن کم‌ترین مقادیر خطای SED و خطای جهت مورد استفاده قرار گرفت. در این روش، مسئله بهینه‌سازی دو هدفه با مجموع وزن‌دار اهداف به مسئله بهینه‌سازی تک هدفه تبدیل شد [۱۶]. در روش پیشنهادی، وزن مربوط به خطای SED و خطای جهت برابر یک در نظر گرفته شد و در روش مجموع وزن‌دار، نرمالایز کردن دو خطا جهت هم مقیاس نمودن آن‌ها انجام شد و سپس مقادیر نرمال این دو خطا با هم جمع شدند. به این ترتیب، اولویت هر نقطه در پنجره متحرک با مقدار تجمیعی که نماینده خطای SED و خطای جهتی در آن نقطه است، محاسبه شد. با در نظر گرفتن همزمان دو هدف کمینه کردن خطای فاصله اقلیدسی همزمانی و خطای تغییر جهت، روش پیشنهادی موفق به حذف نقاط اضافی و حفظ خطای فاصله اقلیدسی همزمانی در یک محدوده قابل قبول شد.

هدف اصلی الگوریتم پیشنهادی، کاهش داده‌های خط سیر با بار محاسباتی کمتر و حفظ نقاط مهم در پیچیدگی خط است. از این رو الگوریتم پیشنهادی از فاصله عمودی به‌عنوان معیار پیچیدگی خط استفاده می‌کند. با استفاده از این معیار برای نقاط با فاصله عمودی بیشتر نسبت به خط واصل نقطه ابتدایی و انتهایی، پنجره اولویت بزرگتری در نظر گرفته می‌شود. سپس با بزرگتر شدن پنجره، تغییرات تفاوت جهت و خطای SED برای هر نقطه نمود بیشتری پیدا می‌کند و باعث می‌شود الگوریتم آن نقطه را با احتمال بیشتری حفظ کند. در حالی که برای نقاط با فاصله عمودی کمتر نسبت به خط واصل نقطه ابتدایی و انتهایی، پنجره اولویت کوچکتری تعیین می‌شود و در نتیجه آن، نقطه مورد نظر با تعداد نقاط کمتری در فرایند فشرده‌سازی الگوریتم پیشنهادی مقایسه شده و زمان فشرده‌سازی خط کاهش می‌یابد. همچنین احتمال حذف این نقاط افزایش می‌یابد. بدین‌صورت نرخ فشرده‌سازی الگوریتم با حفظ نقاط مهم کاهش یافته و از طرف دیگر زمان اجرایی برنامه نیز کم می‌شود.

۴- پیاده‌سازی و ارزیابی

در این بخش داده‌های مورد استفاده در تحقیق (بخش ۴-۱) و نتایج حاصل از مقایسه الگوریتم‌های فشرده‌سازی مذکور تشریح می‌گردند (بخش۴-۲).

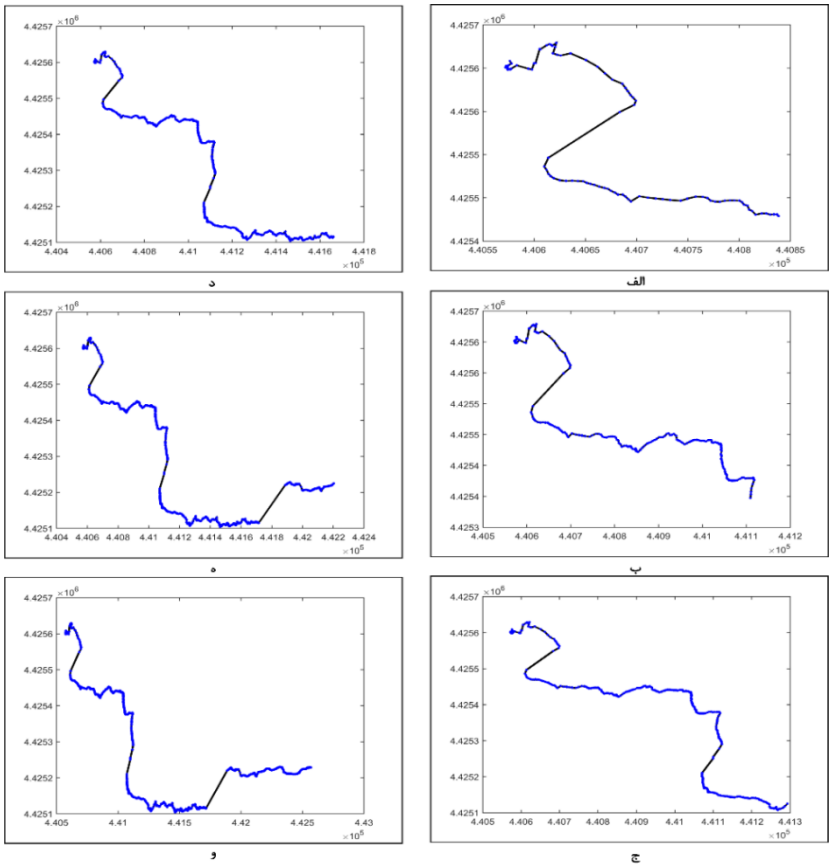
۴-۱- داده‌های مورد مطالعه

خط سیر مورد استفاده در این تحقیق مربوط به مجموعه داده GeoLife است. مجموعه داده‌های GeoLife توسط ۱۸۲ کاربر در بازه زمان پنج ساله از از آوریل ۲۰۰۷ تا آگوست ۲۰۱۲ در شهر پکن جمع‌آوری شده‌اند. هر مسیر GPS از این مجموعه داده، به‌صورت توالی نقاط همراه با اطلاعاتی از جمله طول و عرض جغرافیایی و ارتفاع نقاط می‌باشد. این مجموعه داده شامل ۱۷۶۲۱ مسیر با طول کلی ۱۲۹۲۹۵۱ کیلومتر است. این مسیرها توسط دستگاه‌های مختلف GPS و تلفن‌های همراه دارای GPS با تنوع نرخ نمونه‌برداری ثبت شده‌اند. در این تحقیق، مسیری با طول تقریبا ۲۸۴۷ متر شامل ۶۰۰ نقطه از مجموعه داده GeoLife انتخاب شد. به‌منظور بررسی عملکرد الگوریتم پیشنهادی بر روی خطوط با پیچیدگی متفاوت، خط سیر مورد استفاده به شش زیر مسیر تقسیم شد. برای این منظور با گرفتن صد نقطه ابتدایی مسیر اصلی، زیر مسیر شماره یک تولید شد. به همین ترتیب زیر مسیرهای شماره دو تا شش با در نظرگرفتن

[1] Weighted sum method

[DOI: 10.29252/jgit.6.3.23]

۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰ و ۶۰۰ نقطه ابتدایی از مسیر اصلی تهیه شدند. شکل (۹) این زیرمسیرها را نشان می‌دهد. داده‌های انتخابی توسط GPS در سیستم مختصات جغرافیایی WGS 84 جمع‌آوری شدند. از آنجا که چند الگوریتم مورد استفاده این تحقیق، نیاز به محاسبه سرعت نقاط دارند، روی داده‌های مورد نظر، تبدیل مختصات از سیستم مختصات جغرافیایی به سیستم مختصات دکارتی صورت گرفت.



شکل ۹: خط سیر مورد استفاده تحقیق. الف: زیرمسیری شامل ۱۰۰ نقطه اول، ب: زیرمسیری شامل ۲۰۰ نقطه اول، ج: زیرمسیری شامل ۳۰۰ نقطه اول، د: زیرمسیری شامل ۴۰۰ نقطه اول، ه: زیرمسیری شامل ۵۰۰ نقطه اول و و: زیرمسیری شامل ۶۰۰ نقطه اول مسیر اصلی

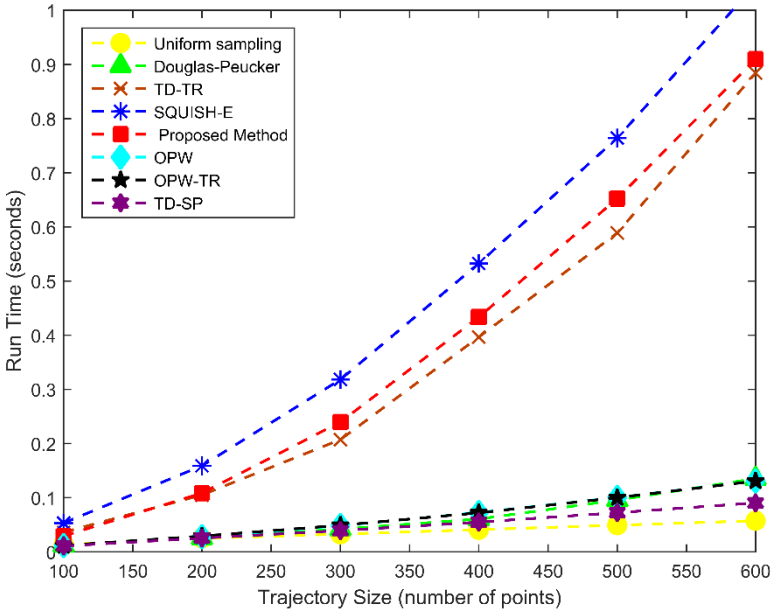
۴-۲- ارزیابی نتایج و بحث

به منظور مقایسه روش‌های کاهش داده خط سیر، الگوریتم‌های ذکر شده بر روی شش زیر مسیر معرفی شده در بخش (۴-۱) پیاده‌سازی شده‌اند. سپس عملکرد این الگوریتم‌ها از نظر معیارهای زمان فشرده‌سازی، نرخ فشرده‌سازی و خطای فاصله اقلیدسی همزمانی ارزیابی و مورد بحث قرار گرفت.

اولین معیار ارزیابی در این تحقیق زمان اجرایی الگوریتم‌ها است. شکل (۱۰) زمان اجرایی هر الگوریتم‌ها را در هریک از زیرمسیرها نشان می‌دهد. در این شکل محور افقی بیانگر اندازه زیر مسیر و محور قائم مدت زمان اجرای الگوریتم است. همانطور که مشاهده می‌شود الگوریتم نمونه‌برداری یکنواخت سریع‌ترین الگوریتم بوده است و در مدت زمان کوتاهی نسبت به سایر الگوریتم، در تمامی زیر مسیرها نقاط خط سیر را کاهش می‌دهد. در حالی که الگوریتم SQUISH-E در تمامی زیرمسیرها به‌دلیل پیچیدگی الگوریتم و در نظر گرفتن فاصله اقلیدسی همزمانی در

[DOI: 10.29252/jgit.6.3.23]

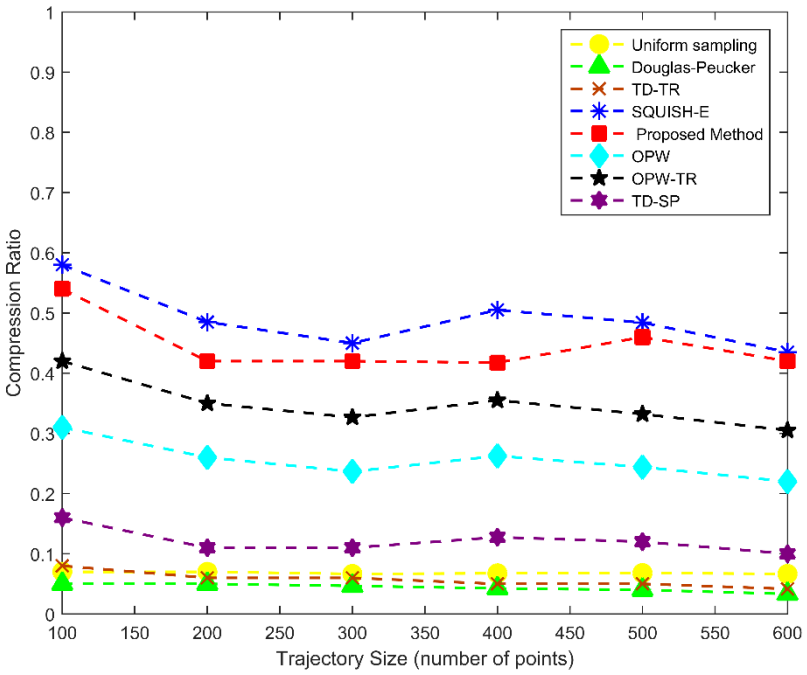
رویکرد کاهش نقاط خط سیر، نسبت به سایر الگوریتم از زمان اجرایی بالایی برخوردار بوده است. همانطور که در شکل (۱۰) مشاهده می‌شود الگوریتم پیشنهادی در مدت زمان کوتاه‌تری نسبت به الگوریتم SQUISH-E، موفق به حذف نقاط اضافه مسیر شده است. علت این مسئله را می‌توان استفاده از پنجره اولویت متغیر و معیار تغییر جهت دانست.



شکل ۱۰: زمان اجرایی هر الگوریتم نسبت به تعداد نقاط مسیر

شکل (۱۱) نرخ فشرده‌سازی الگوریتم‌های مورد استفاده این تحقیق در هر یک از زیر مسیرها را نشان می‌دهد. در این شکل محور افقی بیانگر اندازه زیر مسیر و محور قائم نرخ فشرده‌سازی الگوریتم است. همان‌طور که پیش‌تر ذکر شد، الگوریتم نمونه‌برداری یکنواخت i امین نقطه از مسیر اصلی را ذخیره می‌سازد که در این تحقیق i برابر ۱۵ فرض شده است. این الگوریتم نسبت به سایر الگوریتم نرخ فشرده‌سازی پایین‌تری دارد به‌طوری که زیر مسیر با تعداد نقاط ۶۰۰ را با نرخ فشرده‌سازی ۰٫۰۶۶ فشرده می‌کند. الگوریتم داگلاس پوکر زیر مسیر استفاده شده در روش قبل را با نرخ ۰٫۳۳۳۰ ذخیره می‌کند. همچنین الگوریتم‌های TD-TR، OPW، OPW-TR و TD-SP مسیر با همین تعداد نقاط را با نرخ فشرده‌سازی ۰٫۰۴۲، ۰٫۲۲، ۰٫۳۰۵ و ۰٫۱ فشرده می‌سازد. الگوریتم پیشنهادی نسبت به الگوریتم SQUISH-E نرخ فشرده‌سازی پایین‌تری دارد، نرخ فشرده‌سازی الگوریتم پیشنهادی و الگوریتم SQUISH-E برای زیر مسیر شماره شش به ترتیب برابر ۰٫۴۲و ۰٫۴۳۵می‌باشد. نتایج این تحقیق حاکی از این است که الگوریتم پیشنهادی باعث کاهش نرخ فشرده‌سازی الگوریتم SQUISH-E شده است.

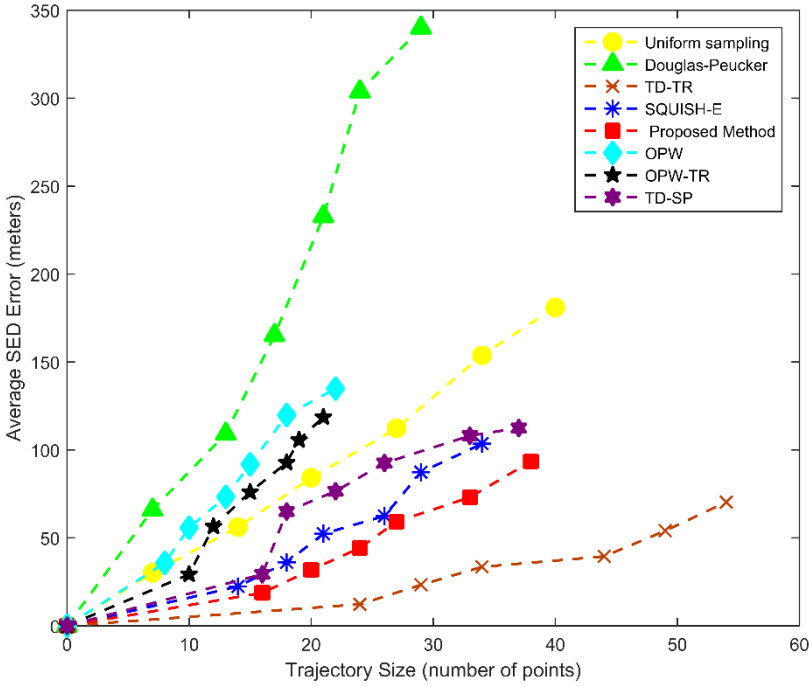
[DOI: 10.29252/jgit.6.3.23]



شکل ۱۱: نرخ فشرده‌سازی هر الگوریتم نسبت به تعداد نقاط مسیر

شکل (۱۲) میانگین خطای فاصله اقلیدسی همزمانی نسبت به تعداد نقاط فشرده شده در هر زیر مسیر را نشان می‌دهد. الگوریتم داگلاس-پوکر به علت استفاده از خطای مکانی بیشترین خطای همزمانی و الگوریتم TD-TR کم‌ترین مقدار خطای را نسبت به سایر الگوریتم‌ها دارد، چرا که از متریک خطای فاصله اقلیدسی همزمانی برای فشرده‌سازی مسیر استفاده می‌کند. الگوریتم OPW-TD جهت تقریب خط سیر، متریک خطای فاصله اقلیدسی همزمانی را در نظر می‌گیرد و مقدار این خطا را نسبت به الگوریتم پنجره متحرک کاهش می‌دهد. الگوریتم SQUISH-E با توجه به اینکه نقاط شکست در مسیر را ذخیره می‌کند، خطای فاصله اقلیدسی همزمانی کمتری دارد. الگوریتم پیشنهادی با در نظر گرفتن همزمان دو خطای فاصله اقلیدسی همزمانی و خطای تغییر جهت، موفق به کاهش خطای فاصله اقلیدسی همزمانی نسبت به الگوریتم SQUISH-E شد.

[DOI: 10.29252/jgit.6.3.23]



شکل ۱۲: خطای فاصله اقلیدسی همزمانی هر الگوریتم نسبت به تعداد نقاط مسیر فشرده

۵- نتیجه‌گیری

با تولید روزافزون داده‌های خط سیر توسط دستگاه‌های تعیین موقعیت GPS، حجم بالای این داده‌ها موجب مشکلاتی در زمینه پردازش، انتقال، دانلود و ذخیره‌سازی داده می‌شود. از این رو الگوریتم‌های فشرده‌سازی مسیر جهت رفع مشکلات مذکور مورد استفاده قرار می‌گیرند. در این تحقیق، هفت الگوریتم فشرده‌سازی هندسی مسیر از جمله الگوریتم نمونه‌برداری یکنواخت، داگلاس پوکر، TD-TR، پنجره متحرک ، OPW-TD، TD-SB و SQUISH-E مورد بحث بررسی قرار گرفتند. مزیت الگوریتم SQUISH-E، برقراری تعادل بین نسبت فشرده‌سازی و خطای فاصله اقلیدسی همزمانی است در حالی که نرخ فشرده‌سازی بالایی نسبت به سایر روش‌ها دارد. در الگوریتم SQUISH-E اندازه پنجره اولویت، که تعیین کننده اولویت هر نقطه در این پنجره است، ابتدا به‌صورت ثابت توسط کاربر تعیین می‌شود و سپس با برقراری شرطی اندازه این پنجره افزایش می‌یابد. در این تحقیق الگوریتم جدیدی به‌منظور بهبود عملکرد الگوریتم SQUISH-E پیشنهاد شد، که در آن اندازه پنجره اولویت متحرک برای هر نقطه متغیر بوده و برای تعیین اولویت هر نقطه از دو معیار فاصله اقلیدسی همزمانی و تغییر جهت استفاده می‌کند.

در پایان الگوریتم پیشنهادی و هفت الگوریتم مذکور روی خط سیری از مجموعه داده GeoLife در نرم‌افزار متلب پیاده‌سازی شدند. به منظور بررسی عملکرد الگوریتم پیشنهادی بر روی خطوط با پیچیدگی متفاوت، خط سیر مورد استفاده به شش زیر مسیر تقسیم شد. سپس عملکرد و کارایی این الگوریتم‌ها با استفاده از معیارهای نرخ فشرده‌سازی، خطای فاصله اقلیدسی همزمانی و زمان اجرای الگوریتم در هر زیر مسیر مورد ارزیابی قرار گرفت. نتایج به دست آمده از این مقایسه حاکی از بهبود نرخ فشرده‌سازی و خطای فاصله اقلیدسی همزمانی الگوریتم پیشنهادی نسبت به الگوریتم SQUISH-E است و همچنین زمان اجرایی روش پیشنهادی کمتر از الگوریتم SQUISH-E می‌باشد.

با بررسی الگوریتم‌های فشرده‌سازی در این تحقیق،

[DOI: 10.29252/jgit.6.3.23]

مشکلات و چالش‌های موجود در فشرده‌سازی مسیر حرکت جسم را می‌توان در موارد زیر خلاصه کرد: ۱) اکثر الگوریتم‌های فشرده‌سازی موجود توجه بیشتری به ویژگی‌های هندسی خط سیر دارند و الگوهای حرکت و ویژگی‌های درونی در مسیرها نادیده گرفته شده است. ۲) اکثر الگوریتم‌های فشرده‌سازی مسیر فعلی نمی‌توانند بعد زمانی و بعد مکانی را به‌طور کامل ترکیب کنند و فقط بعد زمانی را به عنوان بعدی اضافی به بعد مکانی شیء می‌سنجند. ۳) کاربرد عمومی الگوریتم فشرده‌سازی مسیر کم است.

فرایند غنی‌سازی معنایی خط سیرهای مکانی در مدیریت داده‌ها، کاهش تعداد نقاط با حفظ دقت و همچنین شناسایی و تحلیل مکان‌های توقف در خط سیر نقش مهمی دارد. ترکیب اطلاعات معنایی با الگوریتم پیشنهادی می‌تواند مکان‌های مهم از نظر کاربرد با اطلاعات معنایی را حفظ نموده و موجب بهبود عملکرد این الگوریتم شود. ارزیابی کارایی این الگوریتم در کاربردهای مکانی از جمله مدل‌سازی جریان ترافیک، شناسایی معابر مسدود و غیره از جمله پیشنهادات برای مطالعات آتی در این زمینه می‌باشد.

مراجع

[1] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in Proceedings of the 2007 ACM SIGMOD international conference on Management of data, 2007, pp. 593-604.

[2] M. Chen, M. Xu, and P. Franti, "Compression of GPS trajectories," in 2012 Data Compression Conference, 2012, pp. 62-71.

[3] H. Wang, H. Su, K. Zheng, S. Sadiq, and X. Zhou, "An effectiveness study on trajectory similarity measures," in Proceedings of the Twenty-Fourth Australasian Database Conference-Volume 137, 2013, pp. 13-22.

[4] P. Sun, S. Xia, G. Yuan, and D. Li, "An Overview of Moving Object Trajectory Compression Algorithms," Mathematical Problems in Engineering, vol. 2016, 2016.

[5] W. Tobler, "Numerical Map Generalization, Michigan Inter-University Community of Mathematical Geographers," Discussion Paper1966.

[6] R. Bellman, "On the approximation of curves by line segments using dynamic programming," Communications of the ACM, vol. 4, p. 284, 1961.

[7] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," Cartographica: The International Journal for Geographic Information and Geovisualization, vol. 10, pp. 112-122, 1973.

[8] N. Meratnia and R. De By, "A new perspective on trajectory compression techniques," in Proc. ISPRS Commission II and IV, WG II/5, II/6, IV/1 and IV/2 Joint Workshop Spatial, Temporal and Multi-Dimensional Data Modelling and Analysis, 2003.

[9] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, 2001, pp. 289-296.

[10] M. Potamias, K. Patroumpas, and T. Sellis, "Sampling trajectory streams with spatiotemporal criteria," in 18th International Conference on Scientific and Statistical Database Management (SSDBM'06), 2006, pp. 275-284.

[11] J. Muckell, J.-H. Hwang, V. Patil, C. T. Lawson, F. Ping, and S. Ravi, "SQUISH: an online approach for GPS trajectory compression," in Proceedings of the 2nd International Conference on Computing for Geospatial Research & Applications, 2011, p. 13.

[12] J. Muckell, P. W. Olsen Jr, J.-H. Hwang, C. T. Lawson, and S. Ravi, "Compression of trajectory data: a comprehensive evaluation and new approach," GeoInformatica, vol. 18,

[DOI: 10.29252/jgit.6.3.23]

pp. 435-460, 2014.

[13] X. Wu and Z. Cao, "Basic conception, function and implementation of temporal GIS," Earth Science-Journal of China University of Geosciences, vol. 27, pp. 241-245, 2002.

[14] N. Meratnia and A. Rolf, "Spatiotemporal compression techniques for moving point objects," in International Conference on Extending Database Technology, 2004, pp. 765-782.

[15] Y. Zheng and X. Zhou, Computing with spatial trajectories: Springer Science & Business Media, 2011.

[16] K. Deb, "Multi-objective optimization," in Search methodologies, ed: Springer, 2014, pp. 403-449.

[DOI: 10.29252/jgit.6.3.23]

K. N. TOOSI UNIVERSITY OF TECHNOLOGY
FACULTY OF GEODESY AND GEOMATICS ENGINEERING

Journal of Geospatial Information Technology
Vol.6, No.3, Autumn 2018

Data Reduction of Spatio-temporal Trajectories using a Modified Online Compression Algorithm

Afsaneh Nasiri[1], Sanaz Azimi[1], Rahim Ali Abbaspour[*2]

1- MSc, School of Surveying and Geospatial Information Engineering, College of Engineering, University of Tehran.
2- School of Surveying and Geospatial Information Engineering, College of Engineering, University of Tehran

Abstract

With development of mobile devices equipped with a global positioning system, such as smartphones, large amounts of spatial information are generated. These data, which are often stored and modeled as a sequence of spatial locations over time, are called trajectory. The large amount of trajectory data has increased the cost of transferring, storing and processing such data. To overcome these problems, a number of compression algorithms have been proposed for reducing the size of trajectory data. In this paper, seven algorithms including uniform sampling, Douglas Poker, TD-TR, Opening Window, OPW-TR, TD-SB and SQUISH-E algorithms are being discussed and the advantages and disadvantages of these algorithms are investigated as well. The SQUISH-E algorithm can create a balance between the compression rate and the Synchronized Euclidean Distance error, but has a high compression rate than other compression algorithms. To solve mentioned problem, this paper proposed a method for changing the priority window of the SQUISH-E algorithm, which improves the compression rate of this algorithm. In order to evaluate the performance of the proposed method, all algorithms are implemented on six trajectories of varying complexity and compared with each other in terms of criteria such as compression rate, run-time, and concurrency Euclidean distance errors. The results of implementation of the proposed method indicate the improvement of the proposed algorithm at the compression rate, computation time, and Synchronized Euclidean Distance error. In compare to SQUISH-E algorithm, the computation time and compression rate of proposed algorithm is decreased about 130 millisecond and 0.015, respectively.

Key words: Spatio-temporal Trajectories, Compression, SQUISH-E Algorithm, Priority Window.

Correspondence Address GIS Group, school of Surveying and Geospatial Information Engineering, College of Engineering, University of Tehran, Tehran.
Tel : +982188008841
Email : Abaspour@ut.ac.ir

[DOI: 10.29252/jgit.6.3.23]