

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ



دانشگاه اصفهان

دانشکده فنی و مهندسی

گروه مهندسی مکانیک

پایان نامه کارشناسی ارشد رشته‌ی مهندسی مکانیک گرایش طراحی
کاربردی

طراحی و توسعه یک سیستم موقعیت‌سنجی بصری-اینرسی با رویکرد تعقیب
هدف متحرک برای ربات‌های خودمختار

استاد راهنما:

دکتر حسین کریم‌پور

استاد مشاور:

دانشجو:

احمد بابایی بندارتی

شهریور ۱۴۰۲



تعهدنامه اصالت اثر

اینجانب احمد بابایی بندارتی دانشجوی مقطع کارشناسی ارشد رشته مهندسی مکانیک، گرایش طراحی کاربردی متعدد می‌شوم که مطالب مندرج در این پایان نامه و بروندادهای حاصل از آن، دستاوردهای پژوهشی این جانب با اشراف و راهنمایی استاد/استادان دکتر حسین کریمپور است و آن دسته از مطالب این پایان نامه که حاصل تحقیقات دیگران بوده نیز طبق شیوه‌نامه‌های مصوب ارجاع، مستند شده و در فهرست منابع و مآخذ این پژوهش آمده است. ضمناً اظهار می‌دارم که این پایان نامه پیش‌تر برای دریافت هیچ مدرک تحصیلی ارائه نشده است. بدیهی است دانشگاه اصفهان برای خود این حق را قائل است که در صورت احراز و اثبات هر گونه تخلف در این باره، مدرک تحصیلی این جانب را از درجه اعتبار ساقط نماید و ضمن درج موضوع در جاید کثیر الانتشار، کلیه امتیازات و حقوقی را که به موجب آن پس از دوران تحصیل، از آنها بهره‌مند گشته‌ام، از این جانب سلب و به طرف ذی نفع مسترد کند.

- برای رساله‌هایی که با حمایت جزیی مراکز برون‌دانشگاهی تدوین شده است، عبارت زیر تکمیل شود:
این پایان نامه در دانشگاه اصفهان و با حمایت

- برای رساله‌هایی که طی یک قرارداد مشخص، تحت حمایت سازمان یا نهادی تدوین شده است، عبارت زیر تکمیل شود:

این پایان نامه در دانشگاه اصفهان و با حمایت
شده است.

- برای همه رساله‌های تدوین شده در دانشگاه اصفهان عبارت زیر درج شود:
کلیه حقوق مادی و معنوی مترتب بر دستاوردهای مطالعات و نوآوری‌های ناشی از پژوهش در این پایان
نامه در چارچوب آینه نامه مالکیت فکری و تجاری‌سازی دانشگاه تعیین می‌شود.

امضاء

نام و نام خانوادگی دانشجو: احمد بابایی بندارتی

امضاء

نام و نام خانوادگی استاد راهنما: دکتر حسین کریمپور



حوزه معاونت پژوهش و فناوری

دانشگاه اصفهان

دانشکده فنی و مهندسی

گروه مهندسی مکانیک

پایان نامه آقای

احمد بابایی بندارتی

دانشجوی رشته‌ی مهندسی مکانیک گرایش طراحی کاربردی تحت عنوان

طراحی و توسعه یک سیستم موقعیت‌سنجی بصری-اینرسی با رویکرد تعقیب هدف متحرک برای ربات‌های خودمختار

به عنوان بخشی از ملزومات درجه کارشناسی ارشد

در تاریخ ۱۴۰۲/۰۶/۰۴ توسط هیات داوران زیر بررسی و با درجه به تصویب نهایی رسید.

۱- استاد راهنمای پایان نامه دکتر دکتر حسین کریم‌پور راهنما با مرتبه علمی استاد راهنما از
دانشگاه اصفهان

۲- استاد مشاور پایان نامه دکتر نام استاد مشاور با مرتبه علمی استاد مشاور از
دانشگاه /موسسه نام-دانشگاه-یا-موسسه امضا

۳- استاد داور داخل گروه دکتر نام داور داخلی با مرتبه علمی داور داخلی از
دانشگاه /موسسه نام-دانشگاه-یا-موسسه امضا

۴- استاد داور خارج از گروه دکتر نام داور خارجی با مرتبه علمی داور خارجی از
دانشگاه /موسسه نام-دانشگاه-یا-موسسه امضا

مهر و امضای مدیر گروه

اکنون که به یاری خداوند این دوره را به مایان رسانیده‌ام، بر خود واجب میدانم از استاد راهنمای بزرگوارم
جناب آقا دکتر حسین کریمپور به پاس زحمات بی شائبه شان و تحمل کاستی‌های ای جانب در طی انجام
این تحقیق پاکسازی نایم که به سر انجام رسانیدن این مقطع بدون یاری بی دینه ایشان ام کان پذیر
بود.

تقدیم به پدر بزرگوار و مادر مهربانم آن دو فرشته‌ای که از خواسته‌هایشان گذشتند، سختی هارا به جان خرمند و خود را سپری‌بلاسی مشکلات و ناملایات کردند تا من به جایگاهی که اکنون در آن ایستاده ام برسم.

چکیده

امروزه با گسترش دانش در زمینه‌های مختلف علوم و فناوری، حوزه رباتیک با هدف افزایش بهره‌وری و بهینه سازی، در حال توسعه روزافرnon می‌باشد. یکی از مهم‌ترین اهداف این حوزه، کاهش یا حذف کامل دخالت انسانی در انجام فرآیندها می‌باشد؛ چراکه دخالت انسانی در بسیاری از فرآیندها موجب بروز خطاها ناشی از دقت پایین، خستگی و ... می‌گردد. بدین منظور حوزه‌ای جدید تحت عنوان ربات‌های خودمختار با هدف کاهش یا حذف دخالت انسانی به وجود آمده است. معمولاً ساختار سامانه برنامه‌ریز این ربات‌ها از سه بخش اصلی ایجاد نقشه، محلی‌سازی و طراحی مسیر تشکیل شده است که ربات برای هر سه مرحله، نیاز به یک موقعیت‌سنجدی دقیق از محل قرار گیری خود دارد. به این ترتیب موقعیت‌سنجدی دارای اهمیتی ویژه در ربات‌های خودمختار می‌باشد. این موقعیت‌سنجدی می‌تواند به روش‌های مختلف نظری استفاده از GPS، لیزر اسکنر، لیدار، دوربین تکی یا استریو و ... انجام بپذیرد. این روش‌ها همگی دارای مزایای بسیاری می‌باشند ولی برخی به علت گران بودن (لیدار و لیزر اسکنر) و برخی به علت دقت پایین یا مشکلات کالیبراسیون (GPS و دوربین) گزینه مناسبی برای برخی کاربردهای رباتیکی نظری پهپادها نمی‌باشند. به این منظور از روش موقعیت‌سنجدی بصری-اینرسی استفاده می‌شود که از ترکیب یک حسگر سنجش اینرسی (IMU) و یک یا چند دوربین به وجود می‌آید. این روش علاوه بر ارزان و چابک بودن، انعطاف پذیری بالایی برای استفاده در اکثر کاربردهای رباتیک نظری پهپادها را دارد. در این پژوهش، هدف ایجاد یک سیستم موقعیت‌سنجدی بصری اینرسی مبتنی بر فیلتر کالمون چند حالت مقید (MSCKF) می‌باشد که بتوان از آن در کاربرد فرود آمدن پهپاد بر روی سکوی متحرک بهره برد. در این تحقیق، ابتدا مفاهیم ابتدایی مورد نیاز برای توسعه این الگوریتم نظری مدل‌سازی حسگرها، مقدماتی بینایی ماشین، توسعه فیلتر کالمون خطای حالت ارایه و سپس از این مفاهیم برای توسعه روش موقعیت‌سنجدی مربوطه استفاده خواهد شد. در پایان این الگوریتم با کمک کتابخانه‌های معروف OpenCV و Eigen در بستر سیستم عامل ربات (ROS) پیاده‌سازی شده و از آن در کاربرد فرود آمدن پهپاد بر روی سکوی متحرک در قالب شبیه‌سازی استفاده خواهد شد. لازم به ذکر است که برای کاربرد فرود آمدن پهپاد بر روی سکوی متحرک از نرم افزار شبیه‌ساز Gazebo استفاده می‌شود.

کلیدواژه‌ها: سیستم‌های خودمختار - موقعیت‌سنجدی - موقعیت‌سنجدی بصری اینرسی - تعقیب هدف متحرک - سیستم عامل ربات

فهرست مطالب

عنوان	صفحه
فصل اول مقدمه.....	۱
۱-۱ - مقدمه	۱
۱-۲ - موقعیت سنجی.....	۳
۱-۲-۱ - موقعیت سنجی با کمک انکوادر	۴
۱-۲-۲ - موقعیت سنجی با کمک لیزر اسکنر	۵
۱-۲-۳ - موقعیت سنجی با کمک لیدار	۶
۱-۲-۴ - موقعیت سنجی با کمک GPS	۶
۱-۲-۵ - موقعیت سنجی با کمک دوربین (موقعیت سنجی بصری)	۷
۱-۳-۱ - موقعیت سنجی با کمک دوربین استریو	۹
۱-۳-۲ - موقعیت سنجی با کمک حسگر سنجش اینرسی (IMU)	۹
۱-۳-۳ - موقعیت سنجی با کمک دوربین و حسگر سنجش اینرسی (IMU)	۱۰
۱-۳ - موقعیت سنجی بصری-اینرسی	۱۰
۱-۴ - اهداف پژوهش	۱۴
فصل دوم مروری بر مطالعات پیشین	۱۷
۱-۲ - سوابق پژوهشی موقعیت سنجی بصری	۱۷
۱-۲-۱ - سوابق پژوهشی موقعیت سنجی بصری-اینرسی	۲۶
۱-۲-۲ - سوابق پژوهشی موقعیت سنجی بصری-اینرسی	۳۱
فصل سوم طراحی الگوریتم موقعیت سنج بصری-اینرسی	۳۱
۱-۳ - فرآیند انتخاب و طراحی	۳۱
۱-۳-۱ - ساختار واحد اندازه گیری اینرسی	۳۳
۱-۳-۲ - فیلتر کالمن خطای حالت (ESKF)	۳۶
۱-۳-۳ - سینماتیک سیستم در زمان پیوسته	۳۸
۱-۳-۴ - سینماتیک سیستم در زمان گستته	۴۱
۱-۳-۵ - توسعه روابط انتشار فیلتر	۴۳
۱-۳-۶ - توسعه روابط اصلاح فیلتر	۴۵
۱-۳-۷ - مفاهیم مقدماتی بینایی ماشین	۴۶

۴۷.....	۳-۴-۱- شناسایی نقاط ویژه مناسب موقعیت‌سنجی
۵۰.....	۳-۴-۲- تشکیل جریان نوری
۵۵.....	۳-۴-۳- مدلسازی دوربین سوراخ سوزنی
۵۹.....	۳-۴-۴- روش مثلث‌بندی
۶۲.....	۳-۵- پیاده‌سازی فیلتر کالمن چندحالته مقید
۶۳.....	۳-۵-۱- بررسی اجمالی
۶۶.....	۳-۵-۲- بررسی ساختار بردار حالت
۶۷.....	۳-۵-۳- توسعه بخش انتشار حالت
۶۸.....	۳-۵-۴- بخش افزودن حالت
۷۱.....	۳-۵-۵- توسعه مدل رؤیت
۷۷.....	۳-۵-۶- توسعه بخش بروزرسان فیلتر
۷۹.....	فصل چهارم پیاده‌سازی الگوریتم طراحی مسیر فرود
۷۹.....	۴-۱- مدلسازی ریاضی پهپاد (کوادکوپتر)
۸۱.....	۴-۲- الگوریتم طراح مسیر
۸۳.....	فصل پنجم پیاده‌سازی و بررسی نتایج
۸۳.....	۵-۱- معرفی ابزارهای مورد نیاز
۸۴.....	۵-۱-۱- سیستم عامل ربات (ROS)
۸۹.....	۵-۱-۲- کتابخانه پردازش تصویر OpenCV
۹۰.....	۵-۱-۳- کتابخانه حل معادلات جبر خطی Eigen
۹۱.....	۵-۱-۴- شبیه‌ساز Gazebo
۹۲.....	۵-۲- پیاده سازی موقعیت‌سنجی بصری-اینرسی
۹۵.....	۵-۳- پیاده سازی کاربرد فرود آمدن پهپاد بر روی سکوی متحرک
۹۹.....	فصل ششم نتیجه‌گیری و پیشنهادات
۱۰۲	منابع و مأخذ

فهرست شکل‌ها

صفحه	عنوان
۳.....	شکل ۱-۱ انجام همزمان نقشه‌برداری و محلی‌سازی
۵.....	شکل ۱-۲ ساختار حسگر انکوادر نوری و مغناطیسی
۵.....	شکل ۱-۳ لیزر اسکنر
۶.....	شکل ۱-۴ لیدار
۸.....	شکل ۱-۵ نمونه استخراج نقاط ویژه
۹.....	شکل ۱-۶ دوربین استریو
۱۱.....	شکل ۱-۷ ساختار کلی روش کوپل-ضعیف
۱۲.....	شکل ۱-۸ ساختار کلی روش کوپل-محکم
۱۳.....	شکل ۱-۹ ساختار کلی الگوریتم روش موقعیت-سنگی بصری-اینرسی (کوپل-ضعیف)
۱۴.....	شکل ۱-۱۰ تصویری از مسئله فرود آمدن پهپاد بر روی سکوی متحرک در حالت دید کور
۳۴.....	شکل ۱-۱۱ ساختار مفهومی از یک شتاب‌سنجد
۳۴.....	شکل ۱-۱۲ ساختار یک شتاب‌سنجد MEMS
۳۵.....	شکل ۱-۱۳ ساختار یک ژایروسکوپ MEMS
۴۲.....	شکل ۱-۱۴ تاثیر بازه-های زمانی ($t\Delta$) بر روی دقت
۴۸.....	شکل ۱-۱۵ ناحیه‌های مختلف بسته به تغییرات پیکسل‌ها
۵۱.....	شکل ۱-۱۶ بردار حرکت یک جسم در فریم‌های متوالی
۵۱.....	شکل ۱-۱۷ تفاوت بین روش‌های پراکنده و متراکم
۵۳.....	شکل ۱-۱۸ نمایشی از مسئله روزنه
۵۵.....	شکل ۱-۱۹ مدل مربوط به دوربین سوراخ سوزنی
۵۷.....	شکل ۱-۲۰ مقایسه دستگاه مختصات تصویر و دوربین

شکل ۱۱-۳ استفاده از صفحه شترنج برای کالیبراسیون دوربین	۵۸
شکل ۱۲-۳ حالت ایده‌آل مثلثبندی	۵۹
شکل ۱۳-۳ دیاگرام مربوط به روش تقاطع نقطه میانی	۶۰
شکل ۱۴-۳ نمایش چهار داده آخر پنجره کشویی	۶۵
شکل ۱۵-۳ ساختار کلی الگوریتم MSCKF	۶۵
شکل ۱۶-۳ ساختار کلی بخش انتشار حالت	۶۸
شکل ۱۷-۳ ساختار کلی بخش افزایش حالت	۷۱
شکل ۱۸-۳ ساختار کلی بخش روئیت-گر	۷۶
شکل ۱۹-۳ ساختار کلی بخش بروزرسان	۷۸
شکل ۲۰-۴ فرآیند توسعه یک کاربرد رباتیک	۸۰
شکل ۲-۴ ساختار گره‌ای در سیستم عامل ROS	۸۱
شکل ۳-۴ روش‌های ارتباطی بین گره‌های در سیستم عامل ROS	۸۳
شکل ۴-۴ نمای کلی پکیج Rviz	۸۵
شکل ۵-۴ اجرای کاربرد تشخیص چهره با کمک کتابخانه OpenCV	۸۶
شکل ۶-۴ نمایی از شبیه‌سازی محیط صنعتی در Gazebo	۸۸
شکل ۷-۴ تست الگوریتم موقعیت‌سنجی بصری-اینرسی MSCKF	۸۹
شکل ۸-۴ ساختار کلی گره‌های شبیه-سازی	۹۰
شکل ۹-۴ نمایی از شبیه‌سازی کاربرد فرود آمدن پهپاد بر روی سکوی متحرک	۹۱
شکل ۱۰-۴ مسیر طی شده توسط پهپاد برای فرود آمدن بر روی سکوی متحرک	۹۱

فهرست جدول‌ها

عنوان	صفحه
جدول ۱-۲ سوابق پژوهشی مربوط به موقعیت-سنگی بصری ۱۸	
شکل ۲-۲ سوابق پژوهشی مربوط به موقعیت-سنگی بصری-اینرسی ۲۶	

فصل اول

مقدمه

۱-۱ - مقدمه

در سال های اخیر با گسترش دانش در زمینه های مختلف علوم و مهندسی، سطح فناوری و هوشمندی ربات ها بسیار افزایش پیدا کرده است و این ربات ها در حال به کار گرفته شدن در جنبه های مختلف زندگی انسان نظیر صنعت، حمل و نقل، کشاورزی، سلامت و درمان، آموزش، سرگرمی و .. هستند. حوزه رباتیک همواره با پیدایش فناوری ها، تجهیزات و حسگرهای جدید، با هدف افزایش بهره وری و کاهش خطای انجام فرآیند در حال توسعه روزافزون است. یکی از مهم ترین اهداف این حوزه، کاهش (یا حذف کامل) دخالت انسانی در انجام فرآیندها می باشد؛ چرا که دخالت انسانی در بسیاری از فرآیندها موجب بروز خطا های ناشی از دقت پایین، خستگی و ... می گردد. بدین منظور حوزه ای جدید تحت عنوان ربات های خود مختار^۱ (یا به صورت کلی سیستم های خود مختار) با هدف کاهش یا حذف دخالت انسانی به وجود آمده است. بنابر تعریف، ربات های خود مختار، ربات هایی هستند که می توانند وظایف دلخواه را در یک محیط از پیش تعریف نشده، بدون دخالت مداوم انسان انجام دهند [۱]. عموماً زمانی که حرف از ربات ها خود مختار می شود، منظور ربات های مت حرک (که به سکوی ثابت متصل نیستند)

^۱ Autonomous Robots

همانند موبایل ربات^۱، ربات انسان نما^۲، پهپاد بدون سرنشیین^۳ و ... می‌باشد. در سال‌های اخیر، برای ربات‌های خودمختار، الگوریتم‌های^۴ برنامه‌ریزی حرکت مختلفی ارایه شده است که تقریباً همه این الگوریتم‌ها شامل سه بخش مشترک زیر می‌باشند:

۱) ایجاد نقشه^۵: در این بخش، نقشه‌ی محیطی که قرار است ربات در آن فعالیت داشته باشد، با

کمک حسگرهایی نظیر لیزر اسکنر^۶، لیدار^۷، دوربین استریو^۸ و ... استخراج می‌شود و در مراحل بعد مورد استفاده قرار می‌گیرد. لازم به ذکر است که این نقشه می‌تواند به صورت آماده نیز مورد استفاده قرار بگیرد؛ برای مثال در کاربرد فضای باز^۹، می‌توان از نقشه‌های ماهواره‌ای بھرہ برد.

۲) محلی سازی^{۱۰}: در این بخش نیاز است تا بھرہ بردن از داده‌های ورودی حسگرها، موقعیت ربات در نقشه ایجاد شده مشخص شود؛ چراکه برای برنامه‌ریزی حرکت نیاز است تا موقعیت ربات در هر لحظه مشخص باشد.

۳) طراحی مسیر^{۱۱}: در این بخش با داشتن نقشه و موقعیت لحظه‌ای، می‌توان به سمت یک هدف دلخواه حرکت کرد. این هدف دلخواه می‌تواند توسط انسان یا بخش‌های دیگر الگوریتم تصمیم‌گیر تعیین شود.

لازم به ذکر است که در هر سه مورد بالا، داشتن تغییرات مکان ربات به صورت لحظه‌ای مورد نیاز می‌باشد؛ بدین منظور قبل از پیاده‌سازی الگوریتم جهت‌یابی نیاز است تا یک سیستم موقعیت‌سنجی^{۱۲}

¹ Mobile Robots

² Humanoid Robots

³ Unmanned Drones

⁴ Algorithm

⁵ Mapping

⁶ Laser Scanner

⁷ Lidar

⁸ Stereo Cameras

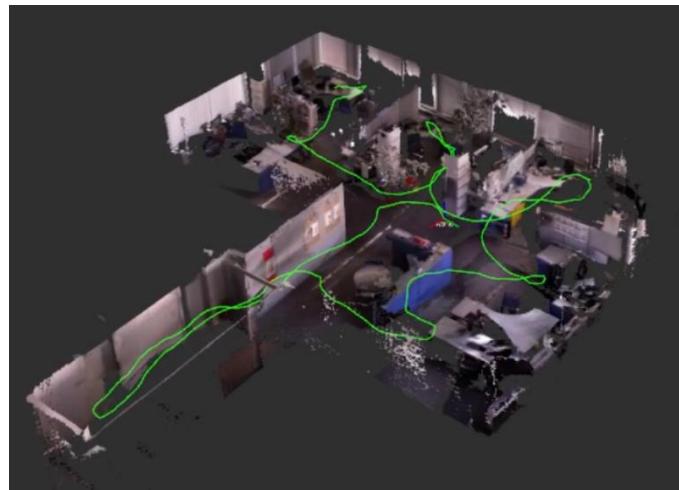
⁹ Outdoor

¹⁰ Localization

¹¹ Path Planning

¹² Odometry

مناسب پیاده‌سازی شود. پایداری و دقت این سیستم موقعیت‌سنجی، تاثیر مستقیم بر صحیح انجام شدن هر سه فرآیند ایجاد نقشه، محلی‌سازی و طراحی مسیر دارد.



شکل ۱-۱ انجام همزمان نقشه‌برداری و محلی‌سازی^۱ [۲]

۱-۲ - موقعیت‌سنجی

معادل انگلیسی موقعیت‌سنجی، odometry است که برگرفته از دو واژه یونانی odos (به معنای مسیر) و metron (به معنای اندازه‌گیری) می‌باشد. بنابر تعریف، موقعیت‌سنجی عبارت است از اندازه‌گیری تغییرات موقعیت ربات با کمک داده‌ی حسگرهای معمولاً یک روش تقریبی است [۳]. در بخش قبلی اهمیت وجود یک سیستم موقعیت‌سنجی دقیق در ربات‌های خودمختار به خوبی مشخص شد. در سال‌های اخیر با توجه به فراهم بودن حسگرهای گوناگون، روش‌های مختلفی برای موقعیت‌سنجی ارایه شده است که برخی از معروف‌ترین آن‌ها عبارتند از:

- موقعیت‌سنجی با کمک انکوادر^۲
- موقعیت‌سنجی با کمک لیزر اسکنر
- موقعیت‌سنجی با کمک لیدار

¹ Simultaneous Localization and Mapping (SLAM)

² Encoder

• موقعیت‌سنجی با کمک GPS^۱

• موقعیت‌سنجی با کمک دوربین

• موقعیت‌سنجی با کمک دوربین استریو^۲

• موقعیت‌سنجی با کمک حسگر سنجش اینرسی

• موقعیت‌سنجی بصری-اینرسی

در ادامه روش‌های موقعیت‌سنجی ارایه شده در لیست بالا بررسی خواهد شد و مزایا، معایب و محدودیت‌های هر کدام ارایه خواهد شد.

۱-۲-۱- موقعیت‌سنجی با کمک انکودر

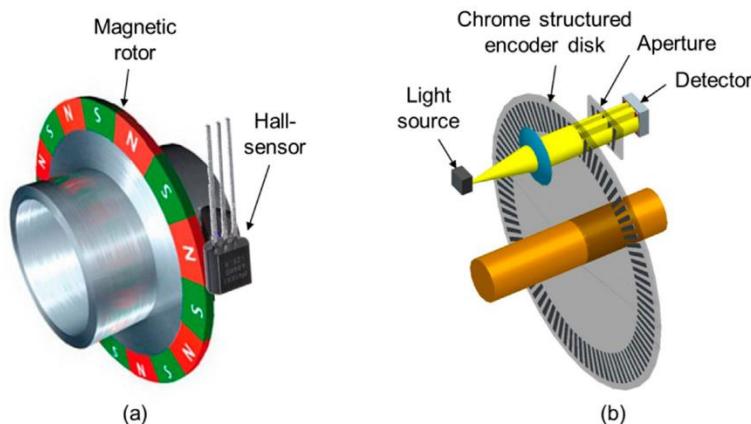
انکودر یک حسگر برای اندازه‌گیری حرکت دورانی چرخ‌ها یک ربات می‌باشد که معمولاً در دو نوع اصلی انکودر نوری^۳ و مغناطیسی^۴ ارایه می‌شود. در انکودر نوری، همانگونه که در شکل شماره ۱-۲ قابل مشاهده است نور بر یک دیسک شیاردار می‌تابد و براساس خاموش و روشن نور، میزان دوران محور سنجیده می‌شود. در مقابل در انکودر مغناطیسی، دیسک دارای قطب‌های مغناطیسی متناوب می‌باشد و با چرخش این دیسک؛ تغییرات میدان مغناطیسی توسط یک حسگر هال اندازه‌گیری می‌شود؛ در نتیجه دوران محور از تناسب نوسانات میدان مغناطیسی محاسبه می‌گردد. سنجش موقعیت با کمک حسگر انکودر یکی از سنتی‌ترین روش‌ها می‌باشد که در آن با کمک داده‌های چرخ‌های ربات تغییرات مکان و جهت ربات تخمین زده می‌شود. از مزایا این روش می‌توان به ارزان بودن و دقیق بودن اندازه-گیری آن بر روی سطوح هموار و غیرلغزندۀ اشاره کرد ولی در صورت غیرهموار یا لغزندۀ بودن سطح، این روش کارایی خود را از دست می‌دهد. همچنین این روش برای ربات‌های انسان‌نما، پهپادها و سایر ربات‌ها که دارای مکانیزم چرخ‌دار نیستند، کارایی ندارد.

¹ Global Positioning System

² Stereo Camera

³ Optical Encoders

⁴ Magnetic Encoders



شکل ۱-۲ ساختار حسگر انکودر نوری و مغناطیسی [۴]

۱-۲-۱ - موقعیت‌سنجی با کمک لیزر اسکنر

یکی از حسگرهایی که می‌توان با آن موقعیت‌سنجی انجام داد، لیزر اسکنر می‌باشد که این سنسور با کمک پالس‌های نوری^۱ فاصله اجسام مختلف را با خود مشخص می‌کند؛ در واقع این حسگر ابتدا در یک صفحه با زوایه‌های مختلف، نور را به اطراف ارسال می‌کند و پس از مدت زمان بسیار کوتاهی (که به زمان پرواز^۲ نور معروف است) بازتاب نور را دریافت می‌کند. با داشتن این مدت زمان پرواز نور، می‌توان فاصله اجسام مختلف را نسبت مرکز سنسور محاسبه کرد. یک نمونه از این لیزر اسکنر در شکل ۳-۱ قابل مشاهده است. از مزایا موقعیت‌سنجی با کمک لیزر اسکنر می‌توان به دقت بالا اشاره کرد. در مقابل قیمت بسیار بالا، سنگین و شکنده بودن جزء معايب اصلی آن می‌باشد.



شکل ۱-۳ لیزر اسکنر مدل [۵] Hokuyo UTM-30LX

^۱ Pulse of Lights

^۲ Time of Flight

۱-۲-۳ - موقعیت‌سنجی با کمک لیدار

حسگر لیدار نیز مشابه لیزر اسکنر می‌باشد، با این تفاوت که در اکثر لیدارهای بجای پالس نوری از امواج رادیویی^۱ استفاده می‌شود (البته برخی از نمونه‌های ارزان قیمت لیدار مبتنی بر همان پالس‌های نوری هستند). از مزایای این موقعیت‌سنجی با کمک لیدار در مقایسه با لیزر اسکنر می‌توان به دقت، بازه اندازه‌گیری بیشتر و مقاوم بودن نسبت به شرایط محیطی (باران، برف، گرد و خاک و ...) اشاره کرد. در مقابل قیمت بالا، وزن بیشتر و ابعاد بزرگتر از معایب آن می‌باشد که استفاده از آن را در اکثر پروژه‌های رباتیکی غیر امکان‌پذیر می‌کند.



شکل ۱-۳-۱ لیدار مدل V1 Yahboom Silan [۶]

۱-۲-۴ - موقعیت‌سنجی با کمک GPS

یکی از راه‌های مرسوم و قدیمی برای موقعیت‌سنجی، استفاده از GPS می‌باشد. در این روش با کمک ارتباط ماهواره‌ای موقعیت تقریبی ربات (یا هر سامانه دیگر) تخمین زده می‌شود. از مزایای اصلی این روش می‌توان به ارزان و قابل حمل بودن آن اشاره کرد. در مقابل عدم کارکرد این حسگر در محیط داخل ساختمان و همچنین محیط‌هایی که ارتباط ماهواره‌ای میسر نیست جزء معایب آن می‌باشد. معمولاً در اکثر کاربردهای رباتیک، از یک حسگر کمکی برای موقعیت‌سنجی در کنار GPS استفاده می‌شود.

^۱ Radio Waves

۱-۲-۵ - موقعیت‌سنجی با کمک دوربین (موقعیت‌ستجی بصری)

دوربین به علت ابعاد کوچک و قیمت مناسب یکی از حسگرهای پرکاربرد در حوزه رباتیک می‌باشد. با کمک داده‌های تصویری دوربین می‌توان کاربردهای مختلفی از جمله شناسایی و دنبال کردن اشیاء^۱، تشخیص چهره^۲، شناسایی و جلوگیری از برخورد با موانع^۳، کنترل کیفیت^۴ و همچین موقعیت‌سنجی بصری پیاده‌سازی کرد. بنابر تعریف موقعیت‌سنجی بصری، به تخمین جابه‌جایی یک سامانه با کمک داده‌های تصویری یک یا چند دوربین، اطلاق می‌شود^[۷]. در سال‌های اخیر روش‌های متنوعی برای موقعیت‌سنجی بصری ارایه شده است که براساس الگوریتم سه دسته اصلی زیر تقسیم می‌شود:

(۱) روش مستقیم^۵

(۲) روش غیرمستقیم^۶ (مبتنی بر ویژگی^۷)

(۳) روش نیمه‌مستقیم^۸ (هیبرید^۹)

در برخی از روش‌های موقعیت‌سنجی بصری، ابتدا با بهره‌گیری از ابزارهای پردازش تصویر، نقاط یا ناحیه‌های ویژه‌ای از هر فریم^{۱۰} استخراج شده (شکل ۱-۴) و از تناظر این نقاط یا ناحیه‌ها در فریم‌های بعدی، تخمین تغییر موقعیت انجام می‌شود که این روش، مبتنی بر ویژگی یا روش غیرمستقیم نامیده می‌شود. در مقابل در برخی روش‌ها، به جای استخراج نقاط یا ناحیه‌های ویژه، از شدت پیکسل^{۱۱}‌های تصویر و تغییرات آن برای تخمین تغییر موقعیت استفاده می‌شود که به این روش‌ها، روش مستقیم گفته می‌شود. برخی از روش‌ها نیز ترکیبی از هر دو روش مستقیم و مبتنی بر ویژگی (غیرمستقیم) می‌باشند، که به آنها روش هیبریدی می‌گویند.

¹ Object Tracking

² Face Detection

³ Obstacle Avoidance

⁴ Quality Control

⁵ Direct Method

⁶ Indirect Method

⁷ Feature-Based

⁸ Semi-Direct Method

⁹ Hybrid

¹⁰ Frame

¹¹ Pixel Intensity



شکل ۱-۴ نمونه استخراج نقاط ویژه [۸]

در روش مستقیم دقت روش تا حدودی بالاتر از روش مبتنی بر ویژگی می‌باشد ولی حجم محاسبات آن نیز بالاتر است. از مزایای اصلی روش موقعیت‌سنجی بصری می‌توان به هزینه پایین، دقت مناسب، وزن پایین و ابعاد کوچک و انعطاف‌پذیری بالا در پیاده سازی کاربردهای مختلف، اشاره کرد. در مقابل برخی از معایب این روش عبارتند از:

- عدم کارایی در شرایط غیرقابل پیش‌بینی تصویر؛ برای موقعیت‌سنجی بصری نیاز است تا تصویرها حاوی بافت^۱ مناسبی باشند. در صورتیکه در دسته‌ای از تصاویر میزان بافت‌ها کم شود، موقعیت‌سنجی بصری بی‌استفاده می‌شود.
- ابهام در تخمین مقیاس: یکی از مشکلات رایج در موقعیت‌سنجی بصری، ابهام در تخمین مقیاس می‌باشد؛ برای مثال در یک تصویر، به علت زاویه دید دوربین ممکن است دو جسم ابعاد یکسان داشته باشند ولی در دنیا واقعی ابعاد جسم دورتر چند برابر جسم نزدیک‌تر باشد. این ابهام مقیاس الگوریتم‌های ارایه شده برای موقعیت‌سنجی بصری را دچار خطای محاسباتی محدود می‌کند (این خطأ برای موقعیت‌سنجی در فضای داخل^۲ ساختمان، قابل صرفه نظر است ولی برای محیط بیرون^۳ از ساختمان اجتناب ناپذیر است).

¹ Textures

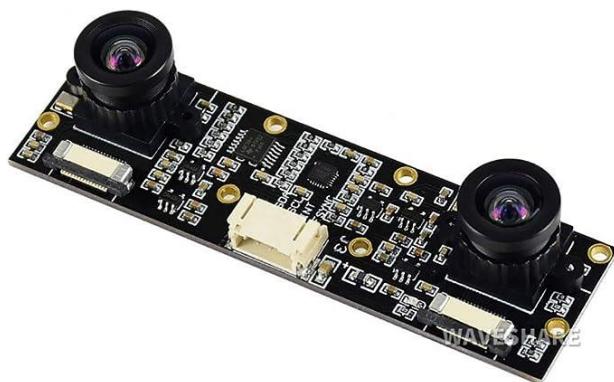
² Indoor

³ Outdoor

بنابراین موقعیت‌سنجی بصری تک دوربینه اگرچه دارای هزینه پایین و انعطاف‌پذیری بالاست ولی به علت مشکل تخمین مقیاس و عدم کارایی در شرایط غیرقابل پیش‌بینی، نمی‌تواند برای کاربردهای رباتیک که نیاز به دقیق قبول دارند، مورد استفاده قرار بگیرد.

۳-۳-۲- موقعیت‌سنجی با کمک دوربین استریو

دوربین استریو در واقع متتشکل از دو دوربین می‌باشد (شکل ۱-۵) که با فاصله‌ای مشخص نسبت به هم قرار می‌گیرند. با کمک دوربین استریو می‌توان علاوه بر مقدار رنگ، برای هر پیکسل عمق را نیز (با کمک مثلث‌بندی^۱ بین نقاط متناظر در تصویر دو دوربین) استخراج کرد. مزیت روش موقعیت‌سنجی بصری استریو (دو دوربینه) در مقایسه با روش تک دوربینه، نداشتن مشکل ابهام تخمین مقیاس و در نتیجه دقیق بیشتر می‌باشد. در مقابل روش استریو حجم محاسبات به مراتب بیشتری دارد و البته کالیبراسیون این روش دارای پیچیدگی‌هایی هست که معمولاً الگوریتم سنجش موقعیت را دچار خطا می‌کند.



شکل ۱-۵ دوربین استریو مدل IMX219

۳-۳-۲- موقعیت‌سنجی با کمک حسگر سنجش اینرسی (IMU)

حسگرهایی سنجش اینرسی (IMU)، متتشکل از شتاب‌سنج خطی^۲ و ژایروسکوپ^۳ می‌باشد که وظیفه سنجش شتاب خطی و سرعت زاویه در سه محور اصلی را برعهده دارد. با کمک داده‌های یک حسگر

¹ Triangulation

² Linear Accelerometer

³ Gyroscope

سنجدش اینرسی (IMU) نیز می‌توان موقعیت‌سنجی تقریبی انجام داد ولی به علت وجود نویز^۱ زیاد این موقعیت‌سنجی دچار خطای فزاینده با زمان می‌شود. در نتیجه علی‌رغم ارزان قیمت بودن، این روش به تنها‌ی برای فرآیند موقعیت‌سنجی در کاربردهای رباتیک مناسب نیست.

۳-۳-۲- موقعیت‌سنجی با کمک دوربین و حسگر سنجش اینرسی (IMU)

فراهم آوردن یک موقعیت‌سنجی دقیق و پایدار نیاز است تا از ترکیب حسگرهای استفاده شود. یکی از این ترکیب‌های موفق که در سال‌های اخیر ارایه شده است، موقعیت‌سنجی با استفاده از دوربین (تکی یا استریو) و حسگر سنجش اینرسی (IMU) می‌باشد که به آن موقعیت‌سنجی بصری-اینرسی می‌گویند. در این نوع از موقعیت‌سنجی از الگوریتم‌های ادغام حسگرهای^۲ برای فراهم آوردن یک موقعیت‌سنجی دقیق و پایدار استفاده می‌شود. در این روش (در صورت استفاده از دوربین تکی) خطای ابهام مقیاس تصویر با کمک داده‌های حسگر سنجش اینرسی (IMU) قابل اصلاح می‌باشد و همچنین مشکل خطاهای فزاینده با زمان مربوط به حسگر سنجش اینرسی (IMU) با کمک داده‌های تصویری قابل جبران است. در بخش‌های بعدی در مورد این روش توضیحات بیشتری ارایه خواهد شد.

۱-۳- موقعیت‌سنجی بصری-اینرسی

مطابق با مطالب بیان شده در بخش قبلی، به موقعیت‌سنجی که در آن از داده‌های دوربین و حسگر سنجش اینرسی (IMU) استفاده شود، روش بصری-اینرسی می‌گویند. این روش ارزان، بهترین انتخاب جایگزین برای حسگرهای GPS و لیدار برای موقعیت‌سنجی می‌باشد که گاه‌آماً در کنار این حسگرهای افزایش دقت موقعیت‌سنجی به کار گرفته می‌شود. این روش به علت استفاده از حسگرهای ارزان قیمت دارای هزینه‌پایین است و با می‌توان آنرا در ابعاد و وزن پایین طراحی کرد. از مزیت‌های دیگر این روش قابلیت استفاده از آن برای محیط داخل و خارج ساختمان و انعطاف پذیری آن در استفاده در ربات‌ها و کاربردهای مختلف است. ساختار الگوریتم موقعیت‌سنجی بصری-اینرسی از سه بخش زیر تشکیل شده است:

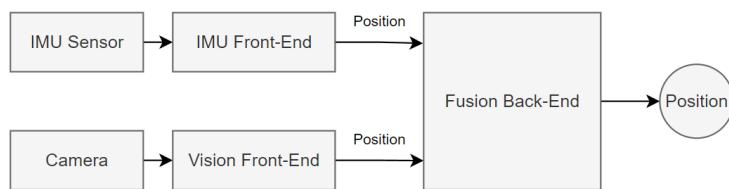
¹ Noise

² Sensor Fusion

- بخش پردازش اینرسی (IMU Front-end)
- بخش پردازش بینایی (Vision Front-end)
- بخش ادغام حسگرها (Sensor Fusion)

داده‌های عددی دو بخش پردازش اینرسی و بینایی می‌توانند با با دو روش کلی زیر با هم ادغام شوند:

- روش کوپل-ضعیف^۱: در این روش بخش‌های پردازش اینرسی و بینایی، دو بخش مجزا و مستقل از هم می‌باشند؛ به عبارت دیگر هر کدام از بخش‌ها به صورت مستقل موقعیت را تخمین زده، سپس خروجی هر دو با هم با کمک الگوریتم‌هایی نظریer فیلتر کالمن^۲، Dempster-Shafer^۳ و ... ادغام شده و یک تخمین موقعیت دقیق‌تر ارایه می‌شود. دیاگرام مربوط به این روش در شکل ۶-۱ ارایه شده است.



شکل ۶-۱ ساختار کلی روش کوپل-ضعیف

- روش کوپل-محکم^۴: در این روش برخلاف روش قبل، دو بخش پردازش اینرسی و بینایی مستقل از یکدیگر نیستند. در این روش در بخش پردازش بینایی ابتدا موقعیت نقاط ویژه از تصویر استخراج می‌شود، سپس داده‌های موقعیت این نقاط ویژه با داده‌های حسگر سنجش اینرسی (IMU) براساس مدل ریاضی خاصی ترکیب شده و تخمین موقعیت انجام می‌دهد. در واقع برخلاف روش قبل موقعیت در یک مرحله تخمین زده می‌شود، که به همین دلیل

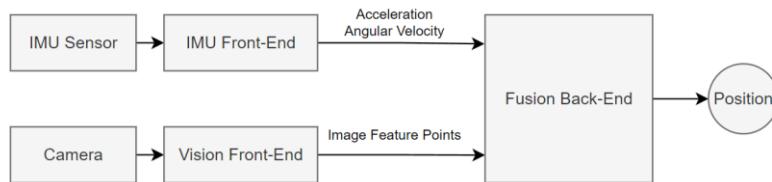
¹ Loosely-Coupled

² Kalman Filter

³ Dempster-Shafer

⁴ Tightly-Coupled

معمولًا روش کوپل-محکم به علت یکبار تخمین کمتر، دقیق‌تر نیز هست. دیاگرام مربوط به این روش نیز در شکل ۷-۱ ارایه شده است.



شکل ۷-۱ ساختار کلی روش کوپل-محکم

در روش موقعیت‌سنجی بصری-اینرسی، بخش پردازش بینایی دارای جزئیات بیشتری می‌باشد.

الگوریتم کلی این بخش از ساختار زیر تشکیل شده است. لازم به ذکر است در برخی از پژوهش‌ها جزئیات ساختاری می‌توانند متفاوت باشند ولی معمولاً اکثر روش‌های دارای مراحل زیر هستند:

۱) دریافت ورودی تصویری: در ابتدا تصویر از یک یا دو دوربین دریافت می‌شود. در صورتیکه دو دوربین استفاده شود، می‌توان با روش بینایی استریو، عمق تصویر را نیز محاسبه کرد.

۲) پیش‌پردازش^۱ تصویر: همانند سایر کاربردهای بینایی ماشین، در ابتدا قبل از انجام پردازش روی تصویر نیاز است تا خطاهای رایج ناشی از دوربین و لنز آن، همانند خطای اثر کی استون^۲، اعوجاج ناشی از لنز^۳ و ... برطرف شود. معمولاً برای این حل مشکلات روش‌های استانداردی ارایه شده است [۱۰] که می‌توان از آنها بهره برد.

۳) استخراج نقاط/ناحیه‌های شاخص: در این مرحله براساس معیارهایی، الگوریتمی ایجاد می‌شود که نقاط/ناحیه مدنظر در یک فریم از تصویر دوربین استخراج شود، سپس نقاط/ناحیه نظیر آن در فریم بعدی نیز استخراج و همبستگی^۴ بین نقاط برقرار می‌شود.

¹ Preprocessing

² Keystone Effect

³ Lens Distortion

⁴ Correlation

۴) تشکیل جریان نوری^۱: بعد از استخراج نقاط/ناحیه‌های شاخص و ایجاد تناظر بین آنها نوبت

به ایجاد بردار جریان نوری می‌رسد. به این منظور از یک الگوریتم معروف با عنوان لوکاس-

کنید^۲ [۱۱] استفاده می‌شود.

۵) ایجاد تخمین گر جابه‌جایی: اکنون با در اختیار داشتن جریان نوری می‌توان به کمک یک مدل

ریاضی این جریان را به جابه‌جایی دوربین تبدیل کرد، به عبارتی دیگر این مدل ریاضی جابه-

جایی دوربین را با توجه به جریان نوری تخمین می‌زند. توجه شود که در روش کوپل-محکم

دیگر این مرحله (مرحله ۵) وجود ندارد و داده‌های خروجی مرحله ۴ به صورت مستقیم با

داده‌های حسگر سنجش اینرسی (IMU) در بخش ادغام سنسورها ترکیب شده و تخمین

موقعیت انجام می‌شود؛ به عبارت دیگر موقعیت با کمک یک مدل ریاضی که ورودی آن جریان

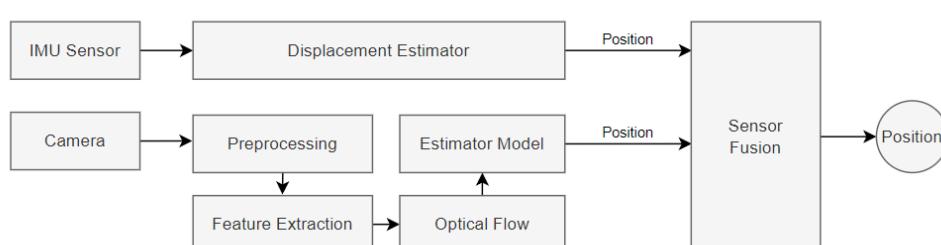
نوری و اطلاعات حسگر سنجش اینرسی (IMU) است، تخمین زده می‌شود.

توجه شود که مراحل ارایه شده برای روش مبتنی بر ویژگی می‌باشد. در روش مستقیم بجای مرحله

شماره ۳ از مراحل بالا، با استفاده از تغییرات شدت تمام پیکسل‌ها، جریان نوری تشکیل می‌شود و بقیه

مراحل مشابه هستند. ساختار کلی الگوریتم روش موقعیت‌سنجی بصری-اینرسی در حالت کوپل-

ضعیف در شکل ۸-۱ قابل مشاهده است.



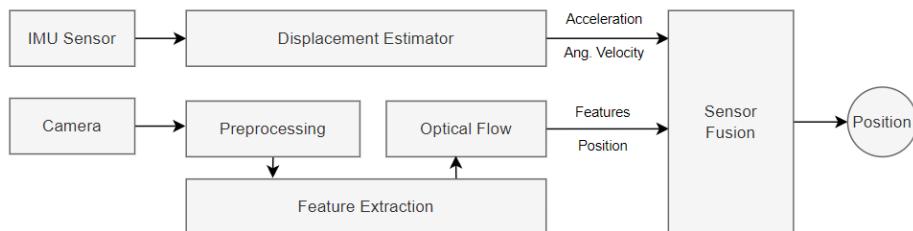
شکل ۸-۱ ساختار کلی الگوریتم روش موقعیت‌سنجی بصری-اینرسی (کوپل-ضعیف)

¹ Optical Flow

² Lucas Kanade

همچنین ساختار کلی الگوریتم موقعیت‌سنجی بصری-اینرسی در حالت کوپل-محکم در شکل ۹-۱

قابل مشاهده است:



شکل ۹-۱ ساختار کلی الگوریتم روش موقعیت‌سنجی بصری-اینرسی (کوپل-محکم)

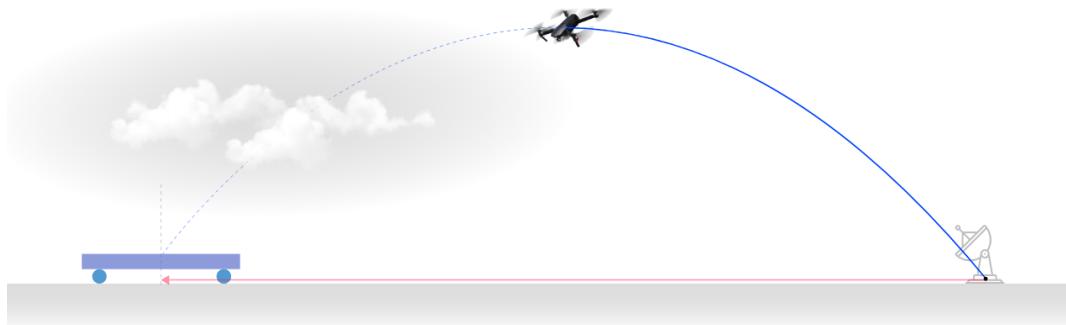
اکنون تا به اینجا، روش موقعیت‌سنجی بصری-اینرسی و اهمیت‌های آن در رباتیک، ارایه شد. همچنین در مورد انواع گوناگون دسته‌بندی و جزئیات الگوریتم آن توضیحاتی بیان شد. در فصل‌های بعدی، بررسی جزئیات بیشتر در مورد تئوری و پیاده سازی این الگوریتم ارایه خواهد شد.

۹-۱-۴ - اهداف پژوهش

یکی از موضوعات پرکاربردی در حوزه رباتیک و پهپادها، تعقیب یک هدف متحرک به صورت کاملا خودکار می‌باشد که در زمینه‌های هدایت و ناوبری ربات‌ها و پهپادها، فرآیندهای نظارتی و ... مورد استفاده قرار می‌گیرد. این موضوع در سال‌های اخیر به دلیل گسترش دوربین‌های باکیفیت و ارزان، با استفاده از دوربین و روش بینایی ماشین انجام می‌پذیرد. در این روش اطلاعات محوری جسم^۱ نظری موقعیت، جهت‌گیری و مساحت تقریبی از تصویر دو بعدی استخراج می‌شود و با کمک این داده‌ها تعقیب هدف متحرک انجام می‌شود. مهمترین چالش در این روش این است که هدف پیوسته باید در پنجره دید دوربین باشد و شرایط برای مشاهده واضح هدف توسط دوربین مهیا باشد ولی اگر این شرایط فراهم نباشد، امکان تعقیب هدف متحرک با کمک دوربین وجود ندارد و باستی از روش دیگر (برای مثال رadar) جهت یافتن موقعیت جسم متحرک بهره برد. در این پژوهش، هدف پیاده‌سازی یک سیستم

¹ Object-Centric Information

موقعیت‌سنجی بصری-اینرسی برای پهپاد بدون سرنشین می‌باشد که پهپاد بتواند با کمک آن، بر روی یک سکوی متحرک فرود بیاید (فرض می‌شود موقعیت این سکوی متحرک را در نظر گیری کرد). به عبارت دیگر در این مسئله موقعیت پهپاد (توسط موقعیت‌سنج بصری-اینرسی) و سکوی متحرک (توسط رادار) نسبت به یک نقطه مبداء معین می‌شود. همچنین فرض می‌شود که دید پهپاد نسبت به سکو کور است و موانع از مشاهده شدن سکوی متحرک توسط پهپاد جلوگیری می‌کند؛ برای مثال سکوی متحرک در فضای مه آلود یا غبارآلود قرار دارد و توسط دوربین قابل مشاهده نیست (همانند شکل ۱۰-۱).



شکل ۱۰-۱ تصویری از مسئله فرود آمدن پهپاد بر روی سکوی متحرک در حالت دید کور

در شکل ۱۰-۱ فرض شده که ابر و مه جلوی دید دوربین پهپاد را گرفته و موقعیت سکوی متحرک صرفاً با کمک رادار قابل تعیین است. بنابراین در جمع‌بندی با توجه به مطالب بیان شده، اهداف این پژوهش به صورت زیر است:

- پیاده‌سازی یک سیستم موقعیت‌سنج بصری که برای کاربرد فرود آمدن پهپاد بر روی سکوی متحرک مناسب باشد. به علاوه از لحاظ حجم محاسبات، قابل اجرا بر روی کامپیوترها با منابع پردازشی محدود (نظیر کامپیوترهای کوچک^۱) باشد.

^۱ Mini-Computer

شبیه‌سازی مثال ذکر شده در شکل ۱۰-۱ با کمک سیستم عامل^۱ ROS و شبیه‌ساز Gazebo

سیستم عامل ROS یک چارچوب متن باز^۲ برای ایجاد پروژه رباتیکی می‌باشد که علاوه بر ارایه ساختار مناسب برای توسعه برنامه، ابزارهای گرافیکی مناسب برای نمایش اطلاعات در اختیار ما قرار می‌دهد. نرم افزار Gazebo نیز یک برنامه متن باز برای شبیه‌سازی سیستم‌های رباتیکی می‌باشد که به امکان شبیه‌سازی فیزیک و دینامیک ربات، سنسورها، عملگرها و ... را می‌دهد. در مورد سیستم عامل ROS و شبیه‌ساز Gazebo در فصل مربوط به پیاده‌سازی توضیحات جامع‌تر داده خواهد شد.

در این فصل در مورد مقدمات ربات و سیستم‌های خودمختار و اهمیت موقعیت‌سنجی در آنها توضیح داده شد. همچنین انواع روش‌های مرسوم موقعیت‌سنجی، بالاخص روش موقعیت‌سنجی بصری-اینرسی بررسی و تحلیل شد. در ادامه این سند، در فصل دوم سوابق پژوهشی مربوطه بررسی خواهد شد. در فصل سوم به بیان تئوری‌ها و روابط لازم برای پیاده‌سازی یک سیستم موقعیت‌سنجی بصری-اینرسی و در فصل چهارم نحوه پیاده‌سازی این الگوریتم‌ها و شبیه‌سازی آن در سیستم عامل ROS و Gazebo بررسی خواهد شد. در نهایت در فصل ششم موارد بیان شده در این پژوهش جمع‌بندی خواهد شد.

¹ Robotic Operation System

² Open-Source

فصل دوم

مروری بر مطالعات پیشین

موضوع موقعیت‌سنگی بصری-اینرسی یکی از مواردی است که در سال‌های اخیر مورد توجه محققان حوزه رباتیک قرار گرفته است و به شیوه‌های مختلف این موضوع اجرا شده است. در این بخش به بررسی برخی از این پژوهش‌ها پرداخته و نوآوری هر کدام به صورت خلاصه بیان خواهد شد. در ابتدا نیاز است تا سوابق پژوهشی مربوطه به موقعیت‌سنگی بصری بررسی شود، چراکه بخش مهمی از ساختار الگوریتم را بخش پردازش بینایی و تحلیل داده‌های تصویری بر عهده دارد. به علاوه اکثر پژوهش‌های مربوط به موقعیت‌سنگی بصری-اینرسی از این پژوهش‌ها استفاده کرده‌اند. پس ابتدا سوابق پژوهشی موقعیت-سنگی بصری و سپس نوع بصری-اینرسی مورد بررسی قرار خواهد گرفت.

۱-۲- سوابق پژوهشی موقعیت‌سنگی بصری

همانگونه که در فصل قبل بیان شد، منظور از موقعیت‌سنگی بصری، تخمین جابه‌جایی یک ربات یا سامانه با کمک داده‌های تصویری یک یا چند دوربین می‌باشد که براساس الگوریتم به مستقیم، غیرمستقیم (مبتنی بر ویژگی) و نیمه مستقیم (هیبرید) تقسیم می‌شود. همچنین این الگوریتم‌ها براساس تعداد و نوع دوربین نیز می‌توانند در دسته‌بندی‌های مجزاء قرار بگیرند. در جدول شماره ۱-۲ می‌توانید فهرست برخی از پژوهش‌های معروف سال‌های اخیر این حوزه را مشاهده نمایید:

جدول ۱-۲ سوابق پژوهشی مربوط به موقعیت‌سنجی بصری

توضیحات	نوع روش	سال	پژوهشگران	مرجع
در این پژوهش برای حرکت ربات بر روی سطوح لغزنه و ناهموار یک سیستم موقعیت‌سنجی بصری ارایه شده است. لازم به ذکر است که سعی شده الگوریتم ارایه شده تا حدودی به لزرشاهی ربات حین حرکت مقاوم باشد.	غیرمستقیم - تک دوربین	۲۰۱۳	Ramon Gonzalez, Francisco Rodriguez, Jose Luis Guzman	[۱۲]
در این مقاله با کمک یک دوربین و یک مسافت یاب لیزری، الگوریتم موقعیت-سنجی ارایه شده است و بر روی یک موبایل ربات، درون حلقه کنترلی روش ارایه شده، امتحان شده است. روش ارایه شده دقیقی همانند حسگر GPS دارد.	غیرمستقیم - تک دوربین	۲۰۱۳	Colin McManus, Paul Furgale, Timothy D. Barfoot	[۱۳]
در این تحقیق موقعیت‌سنجی بصری با کمک الگوریتم های SIFT و SURF بر روی پلتفرم موبایل ربات پیاده سازی شده و نتایج آنها با هم به مقایسه گذاشته شده است.	غیرمستقیم - تک دوربین	۲۰۱۴	Houssem Eddine Benseddk, Oualid Abdel Djekoune, Mahmoud Belhocine	[۱۴]

<p>یک الگوریتم کلی برای استخراج عمق تقریبی با استفاده از داده‌های یک دوربین عادی ارایه شده است. الگوریتم ارایه شده از داده‌های پیکسلی استفاده می‌کند (روش مستقیم) و اگرچه عمق استخراج شده تقریبی است ولی می‌توان از این داده‌ها برای موقعیت‌سنجی استفاده کرد.</p> <p>الگوریتم ارایه شده بر روی یک پردازنده گرافیکی قوی و حافظه بالا نتایج قابل قبولی ارایه می‌کند ولی به نظر می‌رسد که این الگوریتم، برای سیستم با منابع پردازشی پایین (نظیر کامپیوترهای کوچک) مناسب نباشد.</p>	<p>مستقیم</p> <p>-</p> <p>تک دوربین</p>	<p>۲۰۱۴</p>	<p>Matia Pizzoli, Christian Forster, Davide Scaramuzza</p>	<p>[۱۵]</p>
<p>برخلاف روش‌های موقعیت‌سنجی بصری قبلی که در محیط‌های باز یا مقیاس بزرگ از کار می‌افتد، با روش پیشنهاد شده در این پژوهش امکان موقعیت‌سنجی قابل اتکا در محیط‌ها با مقیاس بزرگ فراهم شده است. همچنین برای فیلتر کردن نویز نقشه سه بعدی ایجاد شده توسط روش بصری نیز راه حل ارایه شده است.</p>	<p>مستقیم</p> <p>-</p> <p>تک دوربین</p>	<p>۲۰۱۴</p>	<p>Jakob Engel, Thomas Schops, and Daniel Cremers</p>	<p>[۱۶]</p>

<p>در این مقاله از هر دو روش مستقیم و مبتنی بر ویژگی به صورت ترکیبی بهره گرفته شده است. البته مبنای اصلی روش مستقیم و بهره گرفتن از اطلاعات پیکسل ها می باشد. روش ارایه شده علاوه بر چابک بودن و قابل اجرا بودن بر روی سخت افزارهای پایین رده، دقت مناسبی را نیز فراهم می کند.</p>	<p>نیمه-مستقیم</p> <p>-</p> <p>تک دوربین</p>	<p>۲۰۱۴</p>	<p>Christian Forster, Matia Pizzoli, Davide Scaramuzza</p>	<p>[۱۷]</p>
<p>روشی جدید برای موقعیت سنجی به کمک ORB یک دوربین با بهره گیری از الگوریتم [۱۰] ارایه شد است که به تکانه ها شدید دوربین مقاوم است. ادعا شده است که این روش برای محیط های داخلی و فضای باز مناسب است و همچنین می تواند در کاربرد SLAM به کار گرفته شود. الگوریتم ORB-SLAM ارایه شده در این پژوهش نامیده شده است و در بسیاری از پژوهش های بعدی مورد استفاده قرار گرفته است.</p>	<p>غیرمستقیم</p> <p>-</p> <p>تک دوربین</p>	<p>۲۰۱۵</p>	<p>Raul Mur-Artal, Montiel, Juan D</p>	<p>[۱۸]</p>
<p>در این مقاله از دوربین چشم ماهی بجای دوربین عادی بهره برده شده است. مزیت اصلی این دوربین پوشش ناحیه بیشتر می</p>	<p>مستقیم</p> <p>-</p> <p>تک دوربین</p>	<p>۲۰۱۶</p>	<p>Lionel Heng, Benjamin Choi</p>	<p>[۱۹]</p>

<p>باشد. در این مقاله ابتدا روشی برای کالیبره کردن این نوع دوربین ارایه شده، سپس روش موقعیت سنجی بصری [۱۸] بر روی آن اجرا شده است.</p>	<p>چشم ماهی</p>			
<p>در این پژوهش به جای یک دوربین چشم ماهی از دو دوربین چشم ماهی استفاده شده است و با کمک روش بینایی استریو موقعیت سنجی بصری صورت می‌پذیرد.</p>	<p>مستقیم - دو دوربین چشم ماهی</p>	<p>۲۰۱۷</p>	<p>Peidong Liu, Lionel Heng, Torsten Sattler</p>	<p>[۲۰]</p>
<p>در این پژوهش یک روش موقعیت سنجی ترکیبی مبتنی بر هر دو روش مستقیم و غیر مستقیم ارایه شده است. در این روش یک الگوریتم انتخاب در ابتدای پردازش هر فریم، شرایط تصویر را بررسی می‌کند و براساس شرایط تصویر، یکی از روش‌های مستقیم یا غیر مستقیم را انتخاب می‌کند. در مرحله بعد الگوریتم انتخاب شده بر روی آن فریم اجرا می‌شود. معیار الگوریتم انتخاب، زمان محاسبات و تعداد نقاط ویژه احتمالی می‌باشد.</p>	<p>نیمه-مستقیم - تک دوربین</p>	<p>۲۰۱۷</p>	<p>Jinglun Feng Chengjin Zhang, Bo Sun,</p>	<p>[۲۱]</p>
<p>در این پژوهش ابتدا با استفاده از دو دوربین و با کمک روش مستقیم، نقشه سه</p>	<p>نیمه-مستقیم -</p>	<p>۲۰۱۷</p>	<p>Nicola Krombach, David</p>	<p>[۲۲]</p>

<p>بعدی تقریبی ایجاد می شود، سپس نقاط ویژه مورد نیاز از درون این نقشه سه بعدی انتخاب شده و در نهایت موقعیت سنجی انجام می پذیرد. مزیت اصلی این روش این است که بخش کمی از زمان محاسباتی که بایستی برای یافتن نقاط ویژه هزینه می شد، صرف ساخت نقشه سه بعدی تقریبی می شود. روش ارایه شده علاوه بر دقت مناسب، هزینه پردازشی پایینی دارد.</p>	دو دوربین		Droeschel, Sven Behnke	
<p>در این پژوهش از دوربین‌های رویداد محور برای استخراج جریان نوری بهره برداری شده است. با کمک این دوربین‌ها می توان حرکت اجسام در تصویر را استخراج کرد. این پژوهش، روشی برای موقعیت سنجی بصری مبتنی بر این نوع از دوربین‌ها ارایه کرده است. بیشتر تلاش پژوهندگان بر روی استخراج میدان نوری بوده است.</p>	<p>غیرمستقیم دوربین رویداد محور^۱</p>	<p>- ۲۰۱۷</p>	<p>Alex Junho Lee, and Ayoung Kim</p>	<p>[۲۳]</p>
<p>مشابه پژوهش قبلى [۱۷] می باشد با این تفاوت که الگوریتم ارایه شده قبلى این بار برای حالت چند دوربینی ارایه شده است.</p>	<p>نیمه-مستقیم دو دوربین</p>	<p>- ۲۰۱۷</p>	<p>Christian Forster, Zichao Zhang, Michael</p>	<p>[۲۴]</p>

¹ Event Cameras

<p>طبعتا با افزودن دوربین جدید به مجموعه، دقت موقعیت سنجی افزایش پیدا می کند. البته با بالا رفتن تعداد دوربین ها، حجم محاسبات نیز افزایش می یابد.</p>			Gassner	
<p>در این پژوهش بجای دوربین عادی از یک دوربین RGB-D بهره برداری شده است. این دوربین ها علاوه بر تصویر دو بعدی، عمق (=تصویر سه بعدی) را نیز در اختیار کاربر می گذارند. در روش ارایه شده در این پژوهش ابتدا لبه های اجسام موجود در تصویر سه بعدی استخراج می شوند، سپس با کمک این لبه ها، که در واقع ناحیه های ویژه هستند، موقعیت سنجی به کمک روش غیرمستقیم انجام می شود. عیب الگوریتم ارایه شده، عدم کارکرد در محیط های فضای باز یا بزرگ می باشد.</p>	<p>غیرمستقیم -</p> <p>تک دوربین</p> <p>RGB-D</p>	۲۰۱۸	<p>Yi Zhou, Hongdong Li, Laurent Kneip</p>	[۲۵]
<p>یک روش موقعیت سنجی بصری مستقیم ارایه شده که بجای استفاده از شدت پیکسلی از روش جایگزینی به نام پرداکندگی (Sparse) پیکسلی بهره می</p>	<p>مستقیم -</p> <p>تک دوربینی</p>	۲۰۱۸	<p>Jakob Engel, Vladlen Koltun, Daniel Cremers</p>	[۲۶]

<p>برد. بنابر آزمایش های انجام گرفته این روش می تواند در مقایسه با روش های قبلی علاوه بر دقت بیشتر، منابع پردازشی کمتری را نیز استفاده کند.</p>				
<p>در این پژوهش عملی ابتدا با استفاده از دو دوربین عمق تصویر استخراج شده، سپس با بهره گیری از کتابخانه HBST بخش استخراج ویژگی ها و ایجاد تناظر بین آنها اجرا شده است. کتابخانه HBST یکی از کتابخانه هایی است که در شناسایی بصری محیط^۱ استفاده می شود و این اولین پژوهشی است که از این الگوریتم در حوزه موقعیت سنجی بهره برده است. بنابرنتایج آزمایش ارایه شده، الگوریتم پیشنهادی می تواند در شرایط یکسان (تست بر روی یک دیتاست^۲) دقت بالاتری نسبت به روش ORB-SLAM [۱۸] ارایه کند، در حالیکه از لحاظ پردازش های محاسباتی الگوریتم پیشنهادی سبکتر است. (البته توجه شود که این روش از دو دوربین بهره می برد.)</p>	غیرمستقیم دو دوربینی	-	۲۰۱۸	Dominik Schlegel, Mirco Colosi, Giorgio Grisetti [۲۷]

^۱ Visual Place Recognition

^۲ Dataset

<p>در این پژوهش از یک دوربین تمام نما (۳۶۰ درجه) بجای دوربین عادی استفاده شده است. همچنین از الگوریتم روش ORB-SLAM الهام گرفته شده و یک سیستم موقعیت سنج بصری برای دوربین تمام نما ارایه شده است. استفاده از دوربین تمام نما باعث افزایش دقت و مقاوم بودن سیستم موقعیت سنج شده است.</p>	<p>غیرمستقیم</p> <p>-</p> <p>تک دوربین</p> <p>تمام نما^۱</p>	<p>۲۰۱۹</p>	<p>Shuoyuan Liu, Peng Guo, Lihui Feng</p>	<p>[۲۸]</p>
<p>در این پژوهش از نشانگرهای اعتباری مربعی^۲ برای افزایش دقت استفاده شده است. این نشانگرها در بخش های مختلف محیطی که قرار است ربات با آن تعامل داشته باشد نصب شده و ربات با هر بار عبور و اسکن این نشانگرها می تواند مشخصات مکانی و پردازشی خود را مجدد کالیبره کند. با این روش دقت بلند مدت ربات برای استفاده های طولانی (بیشتر از چند روز) تامین می شود. عیب این روش این است که محیط باید از قبل نشانه</p>	<p>غیرمستقیم</p> <p>-</p> <p>تک دوربین</p>	<p>۲۰۲۰</p>	<p>Rafael Muñoz- Salinas, R. Medina- Carnicer</p>	<p>[۲۹]</p>

¹ Omnidirectional Camera

² Squared Fiducial Markers

گذاری شده باشد.				
در این پژوهش، موقعیت سنجی بصری با استفاده از دوربین رویداد محور انجام می شود. در روش پیشنهاد شده ابتدا نقاط ویژه از تصویر استخراج شده، سپس با کمک دوربین رویدار محور، تغییر مکان این نقاط دنبال شده و جریان نوری تشکیل می شود. با داشتن این جریان نوری دقیق و قابل اتکا، می توان موقعیت سنجی بصری دقیقی را انجام داد. این روش به تکان مقاوم شد است.	غیرمستقیم -	تک دوربین رویداد محور	۲۰۲۱ Weipeng Guan, Peiyu Chen, Yuhan Xie [۳۰]	

۲-۲- سوابق پژوهشی موقعیت سنجی بصری-اینرسی

بنابر تعریفی که در فصل مقدمه ارایه شد، به روشی که در آن از داده های دوربین و حسگر سنجش اینرسی (IMU) استفاده شود، موقعیت سنجی بصری-اینرسی گفته می شود که دسته بندی مشابه روش بصری دارد و همچنین می توان پژوهش ها را براساس کوپل-ضعیف و کوپل-محکم نیز دسته بندی کرد، البته در اکثر پژوهش های سال های اخیر روش کوپل-محکم مورد استفاده قرار گرفته است (به دلیل دقیقی که مراقب می شود). در جدول ۲-۲ می توانید لیست برخی از پژوهش اخیر مربوط به موقعیت سنجی بصری-اینرسی را مشاهده نمایید:

جدول ۲-۲ سوابق پژوهشی مربوط به موقعیت سنجی بصری-اینرسی

توضیحات	نوع	سال	پژوهشگران	مرجع
---------	-----	-----	-----------	------

<p>یک الگوریتم موقعیت سنجی بصری اینرسی برای کاربرد موبایل ربات طراحی و ارایه شده است. روش ارایه شده مبتنی بر ویژگی می باشد و پژوهنده سعی کرده الگوریتم ارایه شده را برای هر دو حالت تک دوربین و دو دوربین اجرا و مقایسه کند. روش ترکیب حسگرها مبتنی بر بهینه سازی می باشد.</p>	<p>غیرمستقیم</p> <p>-</p> <p>تک دوربین</p> <p>دو دوربین</p> <p>-</p> <p>کوپل-محکم</p>	<p>۲۰۱۴</p>	<p>Stefan Leutenegger, Simon Lynen, Michael Bosse</p>	<p>[۳۱]</p>
<p>مهمنترین چالش الگوریتم های موقعیت سنجی بصری-اینرسی، حجم بسیار بالای محاسبات آن می باشد. این محاسبات زمانی که الگوریتم ترکیب حسگرها مبتنی بر بهینه سازی باشد، دو چندان می شود. در صورت بالا بودن حجم محاسبات امکان استفاده از الگوریتم در لحظه (real-time) بدون استفاده از کامپیوترهای بسیار قدرتمند فراهم نمی شود و شاید امکان اجرای این الگوریتم ها بر روی کامپیوترهای کوچک نباشد. در این پژوهش یک روش موقعیت سنجی بصری-اینرسی غیرمستقیم ارایه شده است که</p>	<p>غیرمستقیم</p> <p>-</p> <p>تک دوربین</p> <p>-</p> <p>کوپل-محکم</p>	<p>۲۰۱۷</p>	<p>Christian Forster, Luca Carbone, Frank Dellaert</p>	<p>[۳۲]</p>

علاوه دقت بالا، دارای حجم محاسبات کم می باشد.				
یکی از مهمترین مشکلات در موقعیت سنجی بصری اینرسی، نویز مربوط به حسگر سنجش اینرسی می باشد. در صورتی که از حسگر ارزان قیمت هم استفاده شود، این نویز چند برابر می شود. این پژوهش الگوریتمی مناسب برای حذف این نویزها و جبران خطای ناشی از حسگر ارزان قیمت، ارایه کرده است و الگوریتم ارایه شده را بر روی یک پهپاد اجرا کرده است.	غیرمستقیم - تک دوربین - کوپل-محکم	۲۰۱۷	Jacques Kaiser, Agostino Martinelli, Flavio Fontana	[۳۳]
در این پژوهش روش جدید برای استخراج ویژگی ارایه شده است که علاوه بر بالا بردن دقت و پایداری موقعیت سنجی، حجم محاسبات روش غیرمستقیم را کاهش می دهد. در این روش ابتدا با کمک یک الگوریتم سیک، ناحیه هایی از تصویر که می توانند کاندید مناسبی برای استخراج ویژگی باشند، تعیین می شود. سپس با کمک یک روش دیگر از بین این	غیرمستقیم - تک دوربین - کوپل-محکم	۲۰۱۷	Michael Bloesch, Michael Burri, Sammy Sammy Omari	[۳۴]

<p>ناحیه ها، نقاط ویژه استخراج می شود.</p> <p>همچنین در این پژوهش از روش فیلتر کالمن توسعه یافته برای ترکیب داده های حسگر سنجش اینرسی و این نقاط ویژه استفاده شده است.</p>				
<p>در این پژوهش یک الگوریتم موقعیت سنجی بصری-اینرسی برای پهپاد خود مختار ارایه شده است. همچنین نتایج الگوریتم ارایه شده با برخی از الگوریتم های ارایه شده قبلی مقایسه شده است.</p> <p>لازم به ذکر است که الگوریتم ارایه شده برای محیط های فضاباز مناسب می باشد.</p>	<p>غیرمستقیم</p> <p>-</p> <p>دو دوربین</p> <p>-</p> <p>کوپل-محکم</p>	<p>۲۰۱۸</p>	<p>Ke Sun , Kartik Mohta ,Bernd Pfrommer</p>	<p>[۳۵]</p>
<p>در این پژوهش یک چارچوب کلی برای ترکیب حسگرهای دوربین، سنجش اینرسی و GPS ارایه شده. همچنین چارچوب ارایه شده برای کاربرد سنجش موقعیت خودرو در مسیر شهری به اجرا گذاشته شده است.</p>	<p>مستقیم</p> <p>-</p> <p>دو دوربین</p> <p>-</p> <p>کوپل-محکم</p>	<p>۲۰۱۹</p>	<p>Tong Qin, Shaozu Cao, Jie Pa</p>	<p>[۳۶]</p>
<p>در این پژوهش الگوریتم موقعیت سنجی بصری ارایه شده در پژوهش [۱۸] مبنا قرار</p>	<p>غیرمستقیم</p> <p>-</p>	<p>۲۰۲۰</p>	<p>Carlos Campos, Richard</p>	<p>[۳۷]</p>

<p>گرفته و یک روش موقعیت سنجی بصری اینرسی جدید، مبتنی بر آن ارایه شده است. الگوریتم ارایه در مقایسه با پژوهش های قبلی دارای دقت بالا و در عین حال محاسبات کم می باشد.</p>	<p>تک دوربین - کوپل-محکم</p>		<p>Elvira</p>	
---	------------------------------	--	---------------	--

فصل سوم

طراحی الگوریتم موقعیت‌سنج بصری-اینرسی

در فصل‌های قبل در مورد روش مختلف موقعیت‌سنجی و دلایل برتر بودن روش موقعیت‌سنجی بصری-اینرسی برای کاربردهای پهپادی توضیحات ارایه شد. همچنین برخی از سوابق پژوهشی این موضوع فهرست و بررسی شد. اکنون با دانستن اهمیت این موضوع می‌توان به ساختار و طراحی یک الگوریتم موقعیت‌سنج بصری-اینرسی پرداخت که موضوع این فصل است. در این فصل یک سیستم موقعیت-سنجی بصری-اینرسی مبتنی بر فیلتر کالمن^۱ [۳۸] طراحی و ارایه خواهد شد. در بخش اول این فصل ساختار کلی الگوریتم مدنظر و همچنین نحوه انتخاب حسگرها به صورت کلی بیان خواهد شد. در بخش دوم و سوم واحد اندازه‌گیری اینرسی بررسی و فیلتر کالمن مورد نیاز برای حل نویز و نوسانات آن ارایه خواهد شد. در بخش سوم مفاهیم مورد نیاز از حوزه بینایی ماشین ارایه و در نهایت در بخش چهارم، قسمت ادغام حسگرها و ترکیب داده‌ها بیان خواهد شد.

۱-۳-۱- فرآیند انتخاب و طراحی

در ابتدا مشخص کردن ساختار یک سیستم موقعیت‌سنج بصری-اینرسی نیاز است تا حسگرهای مدنظر خود را انتخاب نماییم و سپس بسته به آنها الگوریتم را چیدمان کنیم. در این پژوهش (با توجه به بودجه

Multi-State Constraint Kalman Filter ^۱

محدود) از یک دوربین تکی (Raspberry Pi Camera) و یک حسگر سنجش اینرسی (MPU9250) ارزان قیمت قسمت استفاده خواهد شد.

مهمنترین مزیت استفاده از دوربین تکی در این ترکیب، در مقایسه با دوربین استریو نداشتن کالیبراسیون‌های چالش برانگیز و حجم محاسبات به مراتب کمتر می‌باشد و همین موضوع سیستم موقعیت‌سنج را قابل اجرا بر روی کامپیوترهای کوچک با ظرفیت محدود می‌کند (اگرچه کمی دقت موقعیت‌سنجی پایین می‌آید). در این پژوهش برای بخش پردازش بینایی (Vision Front-End) از روش غیرمستقیم (مبتنی بر ویژگی) استفاده شده است که این نقاط ویژه از گوشه‌های اجسام موجود در تصویر انتخاب می‌شوند. در بخش پردازش اینرسی (IMU Front-End) مهمترین چالش، استفاده از حسگر سنجش اینرسی ارزان قیمت نویز و نوسان‌های بالای آنهاست که باعث ایجاد خطای فزاینده با زمان در تخمین موقعیت می‌شود. این خطای ترکیب داده‌های دو حسگر با هم در بخش ترکیب داده‌ها قابل حل می‌باشد.

در بخش ادغام حسگر (Sensor Fusion) برای ترکیب داده‌های دوربین و حسگر سنجش اینرسی از روشی مبتنی بر فیلتر کالمن MSCKF استفاده می‌شود که در این روش از یک پنجره کشویی^۱ شامل موقعیت‌های گذشته ربات برای مثلث‌بندی با نقاط ویژه جدید داخل تصویر استفاده می‌شود. این نقاط ویژه مجدد برای بروز کردن موقعیت جدید و پنجره کشویی در مورد بعد مورد استفاده قرار می‌گیرند. لازم به ذکر است که این روش از فیلتر کالمن خطای حالت^۲ جهت تخمین خطای انباشه^۳ شده حسگر سنجش اینرسی (IMU) استفاده می‌کند (بر خلاف فیلتر کالمن توسعه یافته^۴ معمولی که حالت صحیح را تخمین می‌زند). توضیحات بیشتر مربوط به این موارد در بخش مربوط به فیلتر کالمن خطای حالت و ادغام حسگرها ارایه خواهد شد.

^۱ Sliding Window
^۲ Error-State Kalman Filter (ESKF)
^۳ Accumulated Error
^۴ Extended Kalman Filter

۳-۲- ساختار واحد اندازه‌گیری اینرسی

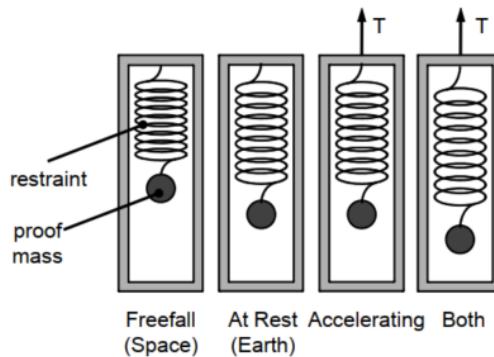
همانگونه که در بخش ۱-۲-۷ بیان شد، حسگر سنجش اینرسی (IMU) از دو بخش شتاب‌سنج خطی و ژایرسکوپ (سنجش‌گر سرعت زاویه‌ای) می‌باشد، البته در برخی از نمونه‌ها یک سنسور سنجش فشار و میدان مغناطیسی نیز قرار می‌گیرد. تا قبل از سال ۱۹۷۰ میلادی از قطعات مکانیکی برای سنجش شتاب خطی و سرعت‌زاویه استفاده می‌شد (معمولاً در صنعت هوا و فضا استفاده می‌شد) که علاوه بر گران قیمت بودن و حجمی بودن این مکانیزم‌ها، برای کارا بودن نیاز به فراهم بودن شرایط محیطی بسیاری بود. در اواخر قرن نوزدهم و اوایل قرن بیستم با پیشرفت در زمینه ریزتراسه‌ها، نمونه الکترونیکی این حسگرها در قالب ریزتراسه ایجاد شد که علاوه بر ارزان و کوچک بودن، قابلیت کارکرد در شرایط محیطی گوناگون را دارد به همین دلیل در کاربردهای مختلف از جمله در تلفن‌ها و تبلت‌های هوشمند، ربات‌ها و ... مورد استفاده قرار می‌گیرند. متاسفانه خروجی اکثر حسگرهای سنجش اینرسی (IMU) به دلیل نقص‌های ایجاد شده در ساختمان یا اجزاء حسگر به هنگام تولید، نمی‌توانند مقدار دقیق پارامترهای فیزیکی را تحويل دهند و معمولاً حاوی خطای می‌باشند، به همین دلیل در مدل‌سازی سامانه این خطاهای را به صورت زیر لحاظ می‌کینم:

$$\mathbf{m}_{measured} = \mathbf{m}_{true} + \mathbf{b} + \mathbf{w} \quad (3-1)$$

که در رابطه بالا \mathbf{m} می‌تواند هر پارامتر خروجی از حسگر (شتاب خطی یا سرعت زاویه‌ای) سنجش اینرسی (IMU) باشد، \mathbf{b} بایاس^۱ حسگر است و \mathbf{w} نویز حسگر می‌باشد. بایاس \mathbf{b} یک مقدار تصادفی است که با خاموش و روشن شدن حسگر مقدار آن تغییر می‌کند ولی در بازه‌ای که حسگر روشن است، این مقدار تقریباً ثابت می‌ماند. نویز \mathbf{w} هم یک مقدار تصادفی است ولی در هر بار اندازه‌گیری اینرسی تصادفی تغییر می‌کند (ربطی به خاموش و شدن حسگر ندارد). سیستم موقعیت‌سنجی بصری-اینرسی در صورت کارا خواهد بود که در روابط این خطاهای تصادفی را در نظر بگیرد. هر حسگر سنجش اینرسی (IMU) حداقل از سه حسگر شتاب‌سنج خطی و از سه حسگر ژایرسکوپ تشکیل شده است (برای اندازه‌گیری پارامترها در سه جهت اصلی). مفهوم کلی شتاب‌سنج خطی در شکل ۱-۳ به نمایش

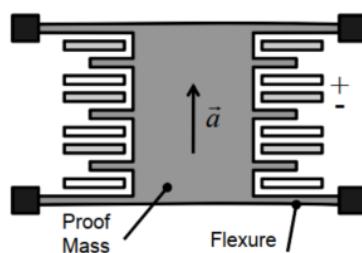
Bias^۱

گذاشته شده است، که در آن با کمک محاسبه نیروی فنر (T) از روی میزان کشیدگی آن می‌توانیم شتاب را در جهات در همان جهت فنر حساب کنیم. برای مثال اگر این شتاب‌سنج به صورت عمودی نسبت به زمین قرار بگیرد، شتاب تقریبی ۹,۸۱ (شتاب جاذبه زمین) قابل محاسبه است.



شکل ۱-۳ ساختار مفهومی از یک شتاب‌سنج [۳۹]

البته در کاربردهای واقعی از ساختار مفهومی استفاده نمی‌شود، بلکه با کمک فناوری سیستم‌های میکرو الکترو-مکانیکال^۱ این مفهوم در قالب ساختار خازن خمس‌پذیر (شکل ۲-۳) در می‌آید. در این ساختار با افزایش شتاب هسته مرکزی تغییر مکان داده و این تغییر مکان منجر به تغییر ظرفیت خازن می‌شود و با کمک اندازه‌گیری ظرفیت خازن (که در الکترونیک دیجیتال به راحتی امکان‌پذیر است) می‌توانیم این شتاب را اندازه‌گیری کینم.



شکل ۲-۳ ساختار یک شتاب‌سنج MEMS [۳۹]

مستقل از اینکه در شتاب‌سنج از چه فناوری استفاده شده باشد، شتاب سنجیده شده توسط این حسگر دارای خطای می‌باشد که به صورت زیر مدل سازی می‌شود:

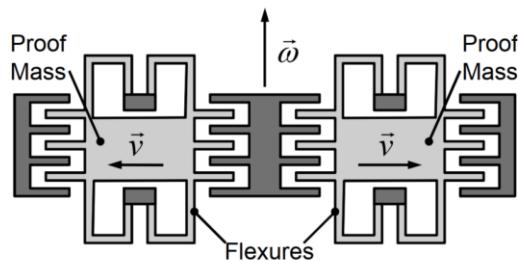
Micro ElectroMechanical Systems (MEMS)^۱

$$\mathbf{a}_m = \mathbf{R}^T(\mathbf{a} - \mathbf{g}) + \mathbf{b}_a + \mathbf{w}_a \quad (3-2)$$

در رابطه بالا \mathbf{a}_m مقدار خوانده شده از سنسور، \mathbf{R} ماتریس دوران شتاب سنج در دستگاه مختصات مرجع می باشد، \mathbf{g} شتاب جاذبه زمین است و \mathbf{b}_a بایاس شتاب سنج می باشد و \mathbf{w}_a نویز آن است. در رابطه بالا شتاب جاذبه (بسته به این سنسور در چه جهتی نسبت به جهت جاذبه قرار گرفته باشد) از شتاب اصلی کم می شود. از رابطه ۲-۳ شتاب اصلی به شکل زیر محاسبه می شود:

$$\mathbf{a} = \mathbf{R}(\mathbf{a}_m - \mathbf{b}_a - \mathbf{w}_a) + \mathbf{g} \quad (3-3)$$

ژایرسکوپ نیز مکانیزم مشابهی با شتاب سنج خطی دارد که در شکل ۳-۳ به نمایش گذاشته شده است. مطابق شکل، در صورت چرخش حسگر حول محور ω فاصله بین صفحات خازن دچار تغییر فاصله می شود و درنتیجه ظرفیت خازن تغییر پیدا می کند. با اندازه گیری این تغییر ظرفیت می توانیم سرعت زاویه ای را محاسبه کنیم.



شکل ۳-۳ ساختار یک ژایرسکوپ MEMS [۳۹]

مشابه شتاب سنج، برای ژایرسکوپ مدل ریاضی اندازه گیری به شکل زیر در می آید:

$$\omega_m = \omega + \mathbf{b}_\omega + \mathbf{w}_\omega \quad (3-4)$$

که در رابطه ω بالا مقدار حقیقی سرعت زاویه ای، ω_m مقدار اندازه گیری شده، \mathbf{b}_ω بایاس و \mathbf{w}_ω نویز می باشد. همچنین سرعت زاویه ای واقعی از رابطه زیر محاسبه می شود:

$$\omega = \omega_m - \mathbf{b}_\omega - \mathbf{w}_\omega \quad (3-5)$$

۳-۳- فیلتر کالمن خطای حالت (ESKF)

در بخش قبل نحوه عملکرد حسگر سنجش اینرسی (IMU) و مدل سازی ریاضی آن بیان شد. در نگاه اول به نظر می‌رشد که می‌توان با دو بار انتگرال گرفتن از شتاب خطی و یک بار انتگرال گرفتن از سرعت زوایه‌ای به تغییرات مکان و جهت‌گیری جسم دست پیدا کرد ولی متاسفانه به علت وجود نویز و خطای این انتگرال‌گیری دچار رانش^۱ داده‌ها می‌شود. نویز موجود در داده‌های حسگر سنجش اینرسی (IMU) با هر بار انتگرال به صورت چند برابری افزایش یافته و ابانته می‌شود و با گذشت زمان این نویز افزایش یافته و تخمین موقعیت عملاً بی‌اعتبار می‌شود. اگرچه می‌توان با استفاده از حسگرهای گران قیمت این چالش را حل کرد ولی با توجه به هزینه بالا حسگرهای بدون نویز، امکان استفاده از آن‌ها در غالب پروژه‌ها امکان‌پذیر نیست. بنابراین نیاز است تا حسگر سنجش اینرسی (IMU) در ترکیب با حسگرهای دیگر (دوربین) فرآیند موقعیت‌سنجی را انجام دهد. با اضافه شدن یک حسگر دیگر، خطای حسگر سنجش اینرسی (IMU) قابل مشاهده^۲ می‌شود که بدین وسیله از رانش داده‌ها جلوگیری می‌شود. بنابراین ما به یک مدل نیاز داریم تا بتوانیم داده‌های حسگر سنجش اینرسی (IMU) را با داده‌های سنسور دیگر (یعنی دوربین) ترکیب کنیم که از فیلتر کالمن خطای حالت بهره می‌گیریم. در فیلتر کالمن خطای حالت، بردار حالت به سه مولفه حالت واقعی^۳، اسمی^۴ و خطای^۵ تقسیم می‌شود که مولفه حالت ترکیبی از مولفه‌های اسمی و خطای می‌باشد. همچنین مولفه خطای کوچک و تقریباً خطی است، در نتیجه ماتریس کواریانس^۶ عدم قطعیت‌ها را در مولفه حالت خطای (Error State) بازتاب می‌دهد (در مقابل در فیلتر کالمن توسعه‌یافته (EKF) عدم قطعیت‌ها در مولفه نامی انکاس داده می‌شود) که با کمک حسگر دوم قابل اصلاح است.

در فیلتر کالمن خطای حالت، فرآیند در ابتدا با ادغام کردن داده‌های حسگر سنجش اینرسی (IMU) در مولفه حالت اسمی (با صرفه نظر از نویز و خطای) آغاز می‌شود. بنابراین خطاهای در مولفه حالت اسمی

Drift	^۱
Observable	^۲
True	^۳
Nominal	^۴
Error	^۵
Covariance Matrix	^۶

انباسه می‌شود. برای حل این مشکل، این خطاهای باید به مولفه حالت خطا انتقال پیدا کند تا با فیلتر با روشی گوسی^۱ یک تخمین از این خطا پیش‌بینی کند (درون فیلتر یک ماتریس کواریانس وجود دارد که نامعینی‌ها را در داخل حالت خطا (Error State) انعکاس می‌دهد و این نامعینی‌ها با گذر زمان افزایش پیدا می‌کند). این فرآیند پیش‌بینی برای هر کدام از داده‌های اندازه‌گیری شده حسگر سنجش اینرسی (IMU) انجام می‌شود تا زمانی که داده‌های حسگر دوم (دوربین) دریافت شود. پس از دریافت داده‌ها از دوربین، خطای انباسته شده قابل مشاهده می‌شود و می‌توان تخمین حالت خطا (Error State) را اصلاح کرد. توجه شود که معمولاً داده‌های دوربین با نرخ کمتری نسبت به حسگر سنج اینرسی (IMU) انتشار پیدا می‌کند، بنابراین خطا در بازه‌های نوسانی کوتاه مدت انباسته و مجدد صفر می‌شود (برخلاف حالت تک حسگری که این خطای انباسته شده واگرا می‌شود). لازم به ذکر است که در فیلتر پس از قابل مشاهده شدن خطا، متوسط آن به مولفه اسمی حالت تزریق شده و خطا برابر صفر قرار داده می‌شود.

از مزایای دیگر روش فیلتر کالمن خطای حالت (ESKF) در مقایسه با فیلتر کالمن توسعه یافته (EKF)، می‌توان به کوچک و خطی بودن حالت خطا (Error State) اشاره کرد که امکان صرفه نظر کردن مشتق دوم خطا را فراهم می‌کند. همچنین در این فیلتر شبیه رشد و انباسته شدن خطا کم می‌باشد و این باعث می‌شود تا بتوان الگوریتم نهایی در نرخ‌های بروزرسانی پایین‌تر اجرا کرد.

در بخش‌های بعدی فیلتر کالمن خطای حالت (ESKF) با جزئیات بیشتر توضیح داده خواهد شد. در بخش‌های ۱-۳-۳ و ۲-۳-۳ شیوه توسعه سینماتیک در زمان پیوسته و گسسته بیان می‌شود و در بخش ۳-۳-۳ و ۴-۳-۳ به ترتیب نحوه طراحی بخش پیش‌بین و اصلاح‌گر فیلتر پرداخته خواهد شد. لازم به ذکر است که فیلتر کالمن خطای حالت (ESKF) ارایه شده در این بخش براساس مرجع [۴۰] پیاده‌سازی شده است.

Gaussian^۱

۳-۳-۳- سینماتیک^۱ سیستم در زمان پیوسته

برای توسعه روابط مربوط به حالت واقعی (True State) از روابط ۳-۳ و ۵-۳ کمک می‌گیریم. براساس

مرجع [۴۰] داریم:

$$\dot{\mathbf{p}}_t = \mathbf{v}_t \quad (3-6-1)$$

$$\dot{\mathbf{v}}_t = \mathbf{a}_t = \mathbf{R}_t(\mathbf{a}_m - \mathbf{b}_{at} - \mathbf{w}_a) + \mathbf{g} \quad (3-6-2)$$

$$\dot{\mathbf{q}}_t = \frac{1}{2} \mathbf{q}_t \otimes \boldsymbol{\omega}_t = \mathbf{q}_t \otimes (\boldsymbol{\omega}_m - \mathbf{b}_{\omega t} - \mathbf{w}_\omega) \quad (3-6-3)$$

$$\dot{\mathbf{b}}_{at} = \mathbf{w}_{ba} \quad (3-6-4)$$

$$\dot{\mathbf{b}}_{\omega t} = \mathbf{w}_{b\omega} \quad (3-6-5)$$

که در روابط ارایه شده در بالا \mathbf{w}_{ba} و $\mathbf{w}_{b\omega}$ بایاس مربوط به شتاب سنج خطی و ژایروسکوپ، \mathbf{R}_t ماتریس دوران و \mathbf{q}_t کواترنیون^۲ می‌باشند. کواترنیون‌ها در سال ۱۸۴۳ توسط ویلیام همیلتون^۳ ابداع شد و یکی از کاربردهای آن توصیف ریاضی دوران در فضای سه‌بعدی می‌باشد. همچین نماد \otimes نشانه ضرب تانسوری^۴ یا ضرب کرونیکر^۵ می‌باشد. همان‌گونه که در مجموعه روابط ۳-۶ قابل مشاهده است، حالت واقعی (True State) شامل نویز و خطأ می‌باشد که نیاز است این نویز و خطأها در مولفه حالت جداگانه-ای ذخیره شوند. به این منظور فرض می‌کنیم که مولفه واقعی حالت (True State)، مولفه اسمی حالت (Nominal State) و مولفه حالت خطأ (Error State) باشد:

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{p}_t \\ \mathbf{v}_t \\ \mathbf{q}_t \\ \mathbf{b}_{at} \\ \mathbf{b}_{\omega t} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{q} \\ \mathbf{b}_a \\ \mathbf{b}_\omega \end{bmatrix}, \quad \delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{p} \\ \delta\mathbf{v} \\ \delta\boldsymbol{\theta} \\ \delta\mathbf{b}_a \\ \delta\mathbf{b}_\omega \end{bmatrix} \quad (3-7)$$

Kinematics ^۱	
Quaternion ^۲	
William Hamilton ^۳	
Tensor Product ^۴	
Kronecker Product ^۵	

توجه شود در رابطه (۳-۷) برای خطای مربوط به جهتگیری از یک تخمین استفاده شده است. با توجه به اینکه تغییرات زاویه‌ای در هر بار اجرای الگوریتم کوچک است است می‌توان براساس [۴۰] از تخمین زیر استفاده کرد:

$$\delta \mathbf{q} \cong \left[\frac{1}{2} \frac{\delta \boldsymbol{\theta}}{\delta \mathbf{x}} \right] \quad (3-8)$$

همان‌گونه که پیشتر هم بیان شد، حالت واقعی (True State) ترکیب است از مولفه اسمی حالت (Nominal State) و مولفه حالت خطا (Error State):

$$\mathbf{x}_t = \mathbf{x} \oplus \delta \mathbf{x} \quad (3-9)$$

که در رابطه بالا منظور از نماد \oplus ترکیب (جمع) مناسب جزء به جزء^۱ می‌باشد. در این جمع برای همه درایه‌های بردارهای رابطه (۳-۸) جمع عادی انجام می‌شود، به عبارت دیگر:

$$\mathbf{p}_t = \mathbf{p} + \delta \mathbf{p} \quad (3-10-1)$$

$$\mathbf{v}_t = \mathbf{v} + \delta \mathbf{v} \quad (3-10-2)$$

$$\mathbf{b}_{at} = \mathbf{b}_a + \delta \mathbf{b}_a \quad (3-10-3)$$

$$\mathbf{b}_{\omega t} = \mathbf{b}_{\omega} + \delta \mathbf{b}_{\omega} \quad (3-10-4)$$

ولی برای درایه‌های مربوط به جهتگیری (کواترنیون و خطای زاویه‌ای) این ترکیت متفاوت است و به شکل زیر تعریف می‌شود:

$$\mathbf{q}_t = \delta \mathbf{q} \otimes \mathbf{q} \quad (3-11)$$

براساس [۴۰] رابطه (۱۱-۳) را می‌توان به صورت معادل زیر نیز محاسبه کرد:

$$\mathbf{R}_t = (\mathbf{I} + [\delta \boldsymbol{\theta}]_{\times}) \mathbf{R} \quad (3-12)$$

که در آن منظور از $[\delta \boldsymbol{\theta}]_{\times}$ یک ماتریس پادمتقارن^۲ می‌باشد که به صورت زیر تعریف می‌شود:

Suitable Component-Wise Composition¹
Skew-Symmetric Matrix²

$$[\delta\theta]_{\times} = \begin{bmatrix} 0 & -\delta\theta_z & \delta\theta_y \\ \delta\theta_z & 0 & -\delta\theta_z \\ -\delta\theta_y & \delta\theta_z & 0 \end{bmatrix} \quad (3-13)$$

اکنون با توجه به داشتن رابطه ترکیب مناسب جزء به جزء که در روابط (۱۰-۳)، (۱۱-۳) و (۱۲-۳) مشخص شد، می‌توان مولفه‌های حالت اسمی و خطرا محاسبه کرد. مولفه اسمی از حذف نویزها به ترتیب زیر بدست می‌آید:

$$\dot{\mathbf{p}} = \mathbf{v} \quad (3-14-1)$$

$$\dot{\mathbf{v}} = \mathbf{a} = \mathbf{R}(\mathbf{a}_m - \mathbf{b}_a) + \mathbf{g} \quad (3-14-2)$$

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \omega = \mathbf{q} \otimes (\omega_m - \mathbf{b}_\omega) \quad (3-14-3)$$

$$\dot{\mathbf{b}}_{at} = 0 \quad (3-14-4)$$

$$\dot{\mathbf{b}}_{\omega t} = 0 \quad (3-14-5)$$

در نهایت مولفه مربوط به حالت خطرا (Error State) نیز با جدا کردن مولفه اسمی مولفه واقعی محاسبه می‌شود و به ترتیب زیر در می‌آید [۴۰]:

$$\dot{\delta\mathbf{p}} = \delta\mathbf{v} \quad (3-15-1)$$

$$\dot{\delta\mathbf{v}} = -[\mathbf{R}(\mathbf{a}_m - \mathbf{b}_a)]_{\times}\delta\theta - \mathbf{R}\delta\mathbf{b}_a - \mathbf{R}\mathbf{w}_a \quad (3-15-2)$$

$$\dot{\delta\mathbf{q}} = -\mathbf{R}\delta\mathbf{b}_\omega - \mathbf{R}\mathbf{w}_\omega \quad (3-15-3)$$

$$\dot{\delta\mathbf{b}}_a = \mathbf{w}_{ba} \quad (3-15-4)$$

$$\dot{\delta\mathbf{b}}_\omega = \mathbf{w}_{b\omega} \quad (3-15-5)$$

از روابط بدست آمده در این بخش برای ایجاد مدل سینماتیک گسسته در زمان (بخش بعدی) استفاده خواهد شد.

۳-۳-۲- سینماتیک سیستم در زمان گسسته

برای پیاده‌سازی کامپیوتری معادلات دیفرانسیل (معادلاتی که به فرم $\dot{x} = f(t, x(t))$ باشند) ارایه شده در بخش قبلی نیاز است تا آن‌ها به فرم گسسته نسبت به زمان تبدیل شوند. برای گسسته سازی بردار حالت از انتگرال گسسته ساز زیر استفاده می‌کنیم:

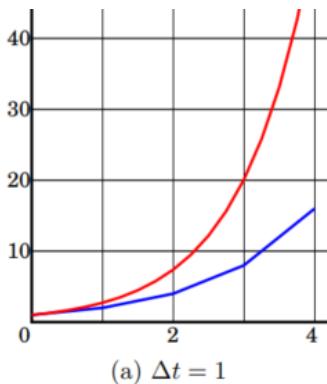
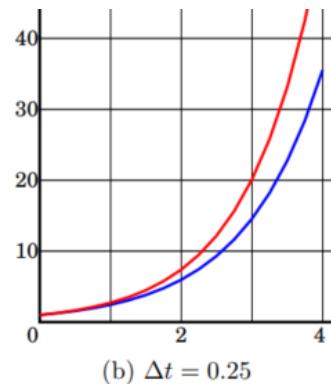
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \int_{k\Delta t}^{(k+1)\Delta t} f(\tau, \mathbf{x}(\tau)) d\tau \quad (3-16)$$

برای محاسبه انتگرال بالا می‌توان از روش‌های تحلیلی یا عددی استفاده نمود. در روش تحلیلی سعی می‌شود با کمک قوانین جبر خطی انتگرال گیری نسبت به τ انجام شود تا مدل ریاضی دقیق بدست آید، متسافانه معادلات در بخش ۱-۳-۱ دارای پیچیدگی‌های بسیار می‌باشد و امکان انتگرال گیری صریح از آن‌ها نیست، پس با استی از روش عددی برای محاسبه انتگرال رابطه (۱۶-۳) استفاده شود. در این پژوهش از یک روش ساده تحت عنوان انتگرال گیری اویلر تک‌گام^۱ استفاده می‌شود [۴۲]. این روش اگرچه ساده و دارای محاسبات بسیار کم است ولی دقت آن پایین برای بازه‌های زمانی (Δt) بلند می‌باشد. خوبی‌خانه اکثر حسگر سنجش اینرسی (IMU) دارای فرکانس بالا می‌باشد و این بازه زمانی (Δt) برای هر بار اندازه‌گیری، عددی بسیار کوچک است. روش اویلر تک‌گام، انتگرال رابطه (۱۶-۳) را به شکل ساده زیر تبدیل می‌کند:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta t f(t_k, \mathbf{x}_k) \quad (3-17)$$

همان‌گونه که از رابطه (۱۷-۳) بر می‌آید هرچه بازه زمانی (Δt) کوچکتر انتخاب شود، دقت بالاتری در تخمین انتگرال فراهم می‌شود. برای مشاهده تاثیر این بازه زمانی (Δt) به شکل ۴-۳ توجه کنید؛ هرچه قدر این بازه زمانی کوچک‌تر باشد، انتگرال گیری عددی به مقدار واقعی آن نزدیک‌تر می‌شود.

^۱One-Step Euler integration

(a) $\Delta t = 1$ (b) $\Delta t = 0.25$

شکل ۳-۴ نتایر بازه‌های زمانی (Δt) بر روی دقت (خط قرمز مقدار واقعی و خط آبی خروجی انتگرال گیری اویلر تگ گام است) [۴۲]

معمولًا فرکانس کاری اکثر حسگرهای سنجش اینرسی (IMU) در بازه ۱۰۰ الی ۱۲۰ هرتز (مقدار Δt تقریباً بین 10^{-4} تا 10^{-3} می‌باشد) که روش انتگرال گیری اویلر تگ گام برای این بازه فرکانسی یک روش مناسب و قابل اتقا است. با استفاده از این روش و به کمک مرجع [۴۰]، حالت گسسته مجموعه روابط (۱۴-۳) که مربوط به مولفه اسمی حالت می‌باشد، به شکل زیر در می‌آید:

$$\mathbf{p}^+ \leftarrow \mathbf{p} + \mathbf{v}\Delta t + \frac{(\mathbf{R}(\mathbf{a}_m - \mathbf{b}_a) + \mathbf{g})\Delta t^2}{2} \quad (3-18-1)$$

$$\mathbf{v}^+ \leftarrow \mathbf{v} + (\mathbf{R}(\mathbf{a}_m - \mathbf{b}_a) + \mathbf{g})\Delta t \quad (3-18-2)$$

$$\mathbf{q}^+ \leftarrow \mathbf{q} \otimes \text{vec2q}((\omega_m - \mathbf{b}_\omega) \Delta t) \quad (3-18-3)$$

$$\mathbf{b}_a^+ \leftarrow \mathbf{b}_a \quad (3-18-4)$$

$$\mathbf{b}_\omega^+ \leftarrow \mathbf{b}_\omega \quad (3-18-5)$$

که در رابطه بالا تابع vec2q بردار چرخش را به کواترنیون تبدیل می‌کند:

$$\alpha = \|\mathbf{v}\| \quad (3-19-1)$$

$$\mathbf{u} = \frac{\mathbf{v}}{\alpha} \quad (3-19-2)$$

$$\text{vec2q}(\mathbf{v}) = \begin{bmatrix} \mathbf{u} \sin(\frac{\alpha}{2}) \\ \cos(\frac{\alpha}{2}) \end{bmatrix} \quad (3-19-3)$$

در روابط بالا \mathbf{v} بردار چرخش به اندازه α رادیان حول محوری به بردار یکه \mathbf{u} می‌باشد. خروجی تابع $vec2q$ کواترنیون متناظر با چرخش بردار \mathbf{v} می‌باشد. همچنین مجموعه روابط (۳-۱۵) که مربوط به مولفه حالت خطأ (Error State) هستند، به طریق مشابه تبدیل به روابط بازگشتی زیر می‌شوند:

$$\delta \mathbf{p}^+ \leftarrow \delta \mathbf{p} + \delta \mathbf{v} \Delta t \quad (3-20-1)$$

$$\delta \mathbf{v}^+ \leftarrow \delta \mathbf{v} + (-[\mathbf{R}(\mathbf{a}_m - \mathbf{b}_a)]_x \delta \boldsymbol{\theta} - \mathbf{R} \delta \mathbf{b}_a) \Delta t + \mathbf{i}_v \quad (3-20-2)$$

$$\delta \boldsymbol{\theta}^+ \leftarrow \delta \boldsymbol{\theta} + \mathbf{R} \delta \mathbf{b}_\omega \Delta t + \mathbf{i}_\theta \quad (3-20-3)$$

$$\delta \mathbf{b}_a^+ \leftarrow \delta \mathbf{b}_a + \mathbf{i}_{ba} \quad (3-20-4)$$

$$\delta \mathbf{b}_\omega^+ \leftarrow \delta \mathbf{b}_\omega + \mathbf{i}_{b\omega} \quad (3-20-5)$$

که در روابط بالا \mathbf{i} تکانه‌های تصادفی هستند (حاصل انتگرال نویز، تکانه می‌شود) که بنابر مرجع [۴۰] با انتگرال‌گیری از ماتریس کواریانس نویزهای \mathbf{w}_a , \mathbf{w}_ω , \mathbf{w}_{ba} و $\mathbf{w}_{b\omega}$ این تکانه‌ها بدست می‌آیند:

$$\mathbf{i}_v = \sigma_a^2 \Delta t^2 \mathbf{I} \quad (3-21-1)$$

$$\mathbf{i}_\theta = \sigma_\omega^2 \Delta t^2 \mathbf{I} \quad (3-21-2)$$

$$\mathbf{i}_{ba} = \sigma_{ba}^2 \Delta t \mathbf{I} \quad (3-21-3)$$

$$\mathbf{i}_{b\omega} = \sigma_{b\omega}^2 \Delta t \mathbf{I} \quad (3-21-4)$$

در تکانه‌های رابطه بالا σ_a , σ_ω , σ_{ba} و $\sigma_{b\omega}$ حاصل مشتق از نویزهای \mathbf{w}_a , \mathbf{w}_ω , \mathbf{w}_{ba} و $\mathbf{w}_{b\omega}$ می‌باشند که از جدول اطلاعات^۱ حسگرهای سنجش اینرسی (IMU) می‌توان این پارامترها را استخراج کرد.

۳-۳-۳- توسعه روابط انتشار فیلتر^۲

فیلتر هر بار که یک اندازه‌گیری جدید از حسگر سنجش اینرسی (IMU) دریافت کند، مولفه‌های اسمی و خطای حالت آن را محاسبه کرده و منتشر می‌کند. در برخی از پژوهش‌ها به علت اینکه فیلتر با استفاده از روابط (۳-۲۰) و (۳-۱۸) به اندازه Δt حالت مرحله بعد را تخمین می‌زند، به بخش انتشار

فیلتر، بخش پیش‌بین^۱ نیز می‌گویند. همچنین همانگونه که پیش‌تر نیز بیان شد، با گذرازمان عدم-قطعیت افزایش یافته که در خطای حالت منعکس می‌شود. این عدم قطعیت‌ها را با ماتریس کواریانس \mathbf{P} نمایش می‌دهیم. بخش انتشار فیلتر به صورت زیر قابل بیان است:

$$\widehat{\delta \mathbf{x}}^+ \leftarrow \mathbf{F}_x \widehat{\delta \mathbf{x}} \quad (3-22-1)$$

$$\mathbf{P}^+ \leftarrow \mathbf{F}_x \mathbf{P} \mathbf{F}_x^T + \mathbf{F}_i \mathbf{Q}_i \mathbf{F}_i^T \quad (3-22-2)$$

در رابطه بالا \mathbf{F}_x ماتریس (زاکوبین^۲) انتقال سیستم با توجه به خطای حالت می‌باشد که از مجموعه روابط (۲۰-۳) حاصل شده است. \mathbf{Q}_i ماتریس کواریانس مربوط به تکانه‌های تصادفی می‌باشد و از روابط (۳-۲۱) برگرفته شده است. ماتریس (زاکوبین) انتقال سیستم با توجه به تکانه‌ها می‌باشد که فقط وظیفه نگاشتن نقاط \mathbf{Q}_i به جایگاه صحیح آنها در ماتریس \mathbf{P} را برعهده دارد. این فرمول‌بندی در مرجع [۴۰] ارایه شده که هر کدام از ماتریس‌های معرفی شده پس مرتب کردن روابطه به شکل زیر در می‌آیند:

$$\mathbf{F}_x = \begin{bmatrix} \mathbf{I} & \mathbf{I}\Delta t & 0 & 0 & 0 \\ 0 & \mathbf{I} & -[\mathbf{R}(\mathbf{a}_m - \mathbf{b}_a)]_x \Delta t & -\mathbf{R}\Delta t & 0 \\ 0 & 0 & \mathbf{I} & 0 & -\mathbf{R}\Delta t \\ 0 & 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{bmatrix} \quad (3-23-1)$$

$$\mathbf{F}_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{I} & 0 & 0 & 0 \\ 0 & \mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{bmatrix} \quad (3-23-2)$$

$$vec2q(\mathbf{v}) = \begin{bmatrix} \mathbf{u} \sin(\frac{\alpha}{2}) \\ \alpha \\ \cos(\frac{\alpha}{2}) \end{bmatrix} \quad (3-23-3)$$

در رابطه (۲-۲۲-۳) عبارت $\mathbf{F}_i \mathbf{Q}_i \mathbf{F}_i^T$ ، دائمانویز و خطرا به ماتریس کواریانس \mathbf{P} اضافه می‌کند که رفتار طبیعی بخش انتشار فیلتر است. در مقابل تازمانی که خطای حالت $\delta \mathbf{x}$ مجدد بر روی صفر تنظیم می‌شود، رابطه (۲-۲۲-۳) به صفر میل می‌کند. لازم به ذکر است که خطای حالت فقط زمان اطلاع فیلتر قابل مشاهده است که در این مورد در بخش بعدی توضیحات بیشتری ارایه خواهد شد.

۴-۳-۳- توسعه روابط اصلاح فیلتر

در بخش قبلی در مورد نحوه پیشینی خطای حالت با کمک داده‌ها حسگر سنجش اینرسی (IMU) توضیحاتی ارایه شد. برای اصلاح این پیش‌بینی‌ها نیاز به یک حسگر اضافه همانند دوربین است تا خط را قابل مشاهده و در نتیجه قابل اصلاح کند. برای مدل‌سازی ریاضی اصلاح‌گر فیلتر فرض می‌کنیم که مدل مشاهده‌گر با h نمایش داده شود (اینتابع مربوط به حسگر دوم، یعنی دوربین می‌باشد و در این بخش توصیف نمی‌شود) و از حالت واقعی (True State) برای انجام عمل مشاهده استفاده کند:

$$\hat{\mathbf{z}} = h(\mathbf{x}_t) + \mathbf{n} \quad (3-24)$$

در رابطه بالا پارامتر \mathbf{n} مربوط به اندازه‌گیری نویز گوسی با ماتریس کواریانس \mathbf{R} می‌باشد. اختلاف بین مقدار مشاهده شده و مقدار مورد انتظار به عنوان باقی‌مانده تعریف می‌شود:

$$\mathbf{r} = \mathbf{z} - \hat{\mathbf{z}} \quad (3-25)$$

اکنون با استفاده از روابط مربوط به بروزرسانی فیلتر کالمن [۴۳] می‌توان خطای حالت را قابل مشاهده کرد:

$$\mathbf{Z} = \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R} \quad (3-26-1)$$

$$\mathbf{K} = \mathbf{P}\mathbf{H}^T\mathbf{Z}^{-1} \quad (3-26-2)$$

$$\widehat{\delta\mathbf{x}} \leftarrow \mathbf{K}\mathbf{r} \quad (3-26-3)$$

$$\mathbf{P} \leftarrow (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}(\mathbf{I} - \mathbf{K}\mathbf{H})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T \quad (3-26-4)$$

در روابط بالا \mathbf{H} ماتریس ژاکوبین مشاهده تابع $(\mathbf{z}, h(\mathbf{x}_t))$ ماتریس کواریانس نوآوری^۱، \mathbf{K} ضریب تقویت فیلتر کالمن و \mathbf{P} یک ماتریس مثبت و متقارن می‌باشد (توضیحات بیشتر در بخش ۵-۳ ارایه خواهد شد). هنگامی که حالت خطای (Error State) مشاهده شد، بایستی به حالت اسمی اضافه شود تا بتوان خطای انباشته شده را اصلاح کرد:

$$\mathbf{x} \leftarrow \mathbf{x} \oplus \delta\mathbf{x} \quad (3-27)$$

^۱Innovation Covariance

که برای هر درایه از بردار حالت، به شکل زیر درمی‌آید:

$$\mathbf{p}^+ \leftarrow \mathbf{p} + \widehat{\delta \mathbf{p}} \quad (3-28-1)$$

$$\mathbf{v}^+ \leftarrow \mathbf{v} + \widehat{\delta \mathbf{v}} \quad (3-28-2)$$

$$\delta \mathbf{q}^+ \leftarrow \text{veq2q}(\widehat{\delta \theta}) \otimes \mathbf{q} \quad (3-28-3)$$

$$\mathbf{b}_a^+ \leftarrow \mathbf{b}_a + \widehat{\delta \mathbf{b}}_a \quad (3-28-4)$$

$$\mathbf{b}_\omega^+ \leftarrow \mathbf{b}_\omega + \widehat{\delta \mathbf{b}}_\omega \quad (3-28-5)$$

در نهایت تا زمانیکه خطای مشاهده و به حالت اسمی (Nominal State) اضافه شود، حالت خطای Error (State) به صفر میل می کند ($\mathbf{0} \leftarrow \delta \mathbf{x}$). در قسمت بعدی مقدماتی درباره بینایی ماشین ارایه خواهد شد تا در نهایت بتوان در قسمت آخر الگوریتم نهایی را اجرا کرد.

۳-۴-۴- مفاهیم مقدماتی بینایی ماشین

امروزه به علت پیشرفت فناوری در زمینه تولید قطعات و حسگرهای الکترونیک، دوربین‌های دیجیتال ارزان قیمت یکی از اجزاء جدایی ناپذیر در اکثر پروژه رباتیکی تبدیل شده و با پیشرفت در حوزه کمک بینایی ماشین، به وفور در کاربردهای مختلف رباتیک (از جمله موقعیت‌سنجی ربات با کمک دوربین) استفاده می‌شوند. بنابر [۴۴]، بینایی ماشین یکی از زیرمجموعه‌های علم کامپیوتر و هوش مصنوعی است که به توسعه و پیشرفت سیستم‌ها و الگوریتم‌ها برای تشخیص و تفسیر تصاویر و ویدئوها می‌پردازد. هدف اصلی در این حوزه، تاکید بر توانایی ماشین‌ها در تشخیص الگوهای اشیاء، چهره‌ها و ویژگی‌های مختلف در تصاویر است. از جمله کاربردهای بینایی ماشین می‌توان به تشخیص اشیاء در تصاویر، تحلیل چهره، خودروهای خودران، تصاویر پزشکی و ... اشاره کرد. همچنین روش بینایی ماشین این امکان را فراهم می‌سازد تا بتوان از تصویر دو بعدی اطلاعات هندسی مربوط به فضای سه بعدی را تخمین زد و از این اطلاعات در عمل موقعیت‌سنجی بهره برد. در روش موقعیت‌سنجی بصری اینرسی از این اطلاعات هندسی جهت اصلاح خطای مربوط به حسگر سنجش اینرسی (IMU) نیز استفاده می‌شود. در این بخش مباحث مقدماتی و مورد نیاز برای پیاده‌سازی یک الگوریتم موقعیت‌سنج بصری-اینرسی

ارایه شده است؛ در ابتدای این قسمت (بخش ۱-۴-۳) در مورد نقاط ویژه و روش استخراج آنها ارایه شده است که در بخش ۲-۴-۳ در روش دنبال کردن این نقاط ویژه و نحوه تخمین زدن حرکت آنها بررسی شده است. در بخش ۳-۴-۳ ساختار و شیوه مدل‌سازی دوربین‌های سوراخ سوزنی^۱ بررسی شده و در نهایت در بخش ۴-۴-۳ روش مثلث‌بندی بررسی شده است.

۱-۴-۳- شناسایی نقاط ویژه مناسب موقعیت‌سنگی

منظور از نقاط ویژه در روش موقعیت‌سنگی بصری-اینرسی، نقاطی هستند که نسبت به نقاط همسایه خود متفاوت‌اند و تقریباً می‌توان آنها را در فریم‌های مختلف تصویر مجدد شناسایی کرد. نقاط ویژه علاوه بر موقعیت‌سنگی بصری-اینرسی در حوزه دیگر از جمله واقعیت افزوده، نقشه برداری و ... کاربرد فراوان دارد. برای مثال در کاربرد واقعیت افزوده^۲ برای اینکه بتوانیم اجسام سه بعدی را در داخل تصاویر دنیای واقعی جاسازی کنیم، نیاز است تا محل برخی نقاط ویژه داخل تصویر فراهم باشد، برای مثال این نقاط ویژه می‌تواند مختصات قرارگیری برچسب‌های^۳ واقعیت افزوده باشد.

در الگوریتم موقعیت‌سنگ بصری اینرسی نیاز است تا در ابتدای این نقاط ویژه به شیوه استخراج شود و با موقعیت آن فریم بعدی تصویر تطابق داده شود؛ به عبارت دیگر در این بخش از الگوریتم دو جزء اصلی داریم:

- شناساگر نقاط ویژه: وظیفه این جزء شناسایی و ارایه موقعیت نقاطی از تصویر است که شرایط کافی برای ویژه بودن را داشته باشند، به عبارت دیگر این نقاط متمایز در فریم‌های متوالی تصویر قابل شناسایی مجدد باشند.
- توصیف کننده نقاط ویژه: بعد از شناسایی نقاط ویژه در فریم‌های مختلف تصویر نیاز است تا بین این نقاط تناظر برقرار شود که این وظیفه جزء توصیف کننده می‌باشد. در این جزء الگوریتم با مقایسه پیکسل‌های موجود در همسایگی نقاط ویژه در دو فریم متوالی، سعی می-

کند تا بین نقاط مشابه تناظر برقرار کند (در این جزء معمولاً نقاطی که احتمال ایجاد تناظر

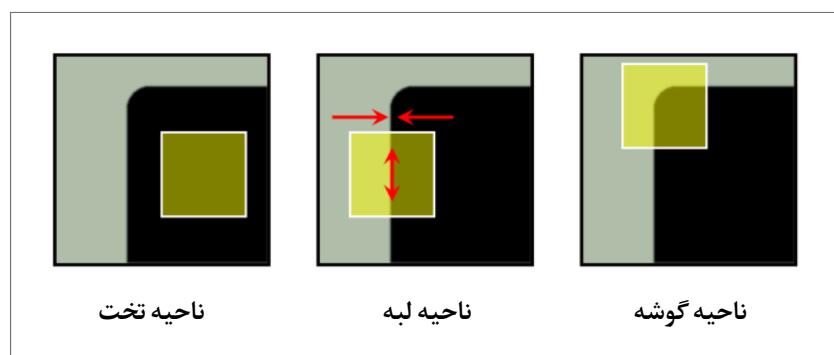
بین آنها با خطأ روبرو می‌شود، از لیست حذف می‌شوند).

اگرچه نقاط ویژه می‌توانند گوشه‌ها^۱، لبه‌ها^۲، لکه‌ها^۳ و ... باشند ولی در این پژوهش به علت حجم محاسبات کم و سهولت در توسعه پارامترها، از گوشه‌ها به عنوان نقاط ویژه استفاده خواهد شد. برای شناسایی گوشه‌های موجود در یک تصویر الگوریتم‌های متنوعی ارایه شده است که روش شناسایی گوشه هریس^۴ [۴۵] یکی از معروف‌ترین آنها می‌باشد. در این روش یک پنجره کوچک از پیکسل‌ها انتخاب می‌شود و بر روی تصویر از دو جهت (در جهت x و y) حرکت داده می‌شود و تغییرات سنجیده پیکسل‌ها سنجیده می‌شود. براساس این تغییرات می‌توان پیش‌بینی کرد نقطه مورد بررسی چه حالتی

دارد:

- اگر تغییرات پیکسل‌ها ناچیز باشد ناحیه تحت^۵ (بدون تغییر) است.
- اگر تغییرات پیکسل در یک جهت (x یا y) زیاد باشد ناحیه لبه است.
- اگر تغییرات پیکسل در هر دو جهت (x و y) زیاد باشد، ناحیه گوشه است.

حالتهای ذکر شده بالا را می‌توانید به صورت مصور در شکل ۵-۳ مشاهده کنید:



شکل ۵-۳ ناحیه‌های مختلف بسته به تغییرات پیکسل‌ها [۴۶]

Corners	^۱
Edges	^۲
Blobs	^۳
Harris Corner Detector	^۴
Flat	^۵

برای اندازه‌گیری این تغییرات پیکسل از روش وزنی جمع مربعی تغییرات^۱ استفاده پیشنهاد می‌شود [۴۷]. در این روش از شدت پیکسل^۲ (شدت برابر است مقدار روشنایی هر پیکسل تصویر در مقیاس خاکستری^۳) برای سنجیدن تغییرات بین پیکسل‌ها استفاده می‌شود:

$$S(x, y) = \sum_{x,y} w(x, y)[I(x + u, y + v) - I(x, y)]^2 \quad (3-29)$$

در رابطه بالا $S(x, y)$ خروجی صاف شده پنجره متوجه در نقطه y ، x ، $w(x, y)$ وزن، $I(x, y)$ و پارامترهای u و v ابعاد پنجره متوجه است. رابطه بالا را می‌توان با کمک بسط تیلور^۴ به رابطه تقریبی زیر تبدیل کرد (I_x و I_y مشتقات شدت در راستای x و y هستند):

$$S(x, y) \approx \sum_{x,y} w(x, y)[uI_x + vI_y]^2 \quad (3-30)$$

رابطه بالا را می‌توان به فرم ماتریسی زیر تبدیل کرد:

$$S(x, y) \approx [u \quad v] \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix} \quad (3-31)$$

که در رابطه بالا ماتریس مشتقات \mathbf{M} به صورت زیر تعریف می‌شود:

$$\mathbf{M} = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (3-32)$$

تغییرات ماتریس مشتقات \mathbf{M} برای ناحیه‌ها با تغییرات مختلف (تخت، لبه و گوشه) متفاوت است. برای سنجیدن این تغییرات می‌توان از مقادیر ویژه ماتریس M بهره برد که در واقع نمایانگر تغییرات در راستای x و y می‌باشد:

- اگر مقادیر ویژه هر دو کوچک باشند، ناحیه تخت است.
- اگر یکی از مقادیر ویژه کوچک و دیگر بزرگ باشد، ناحیه لبه است.
- اگر هر دو مقدار ویژه بزرگ باشند، ناحیه گوشه است.

Sum of Squared Differences^۱
Pixel Intensity^۲
Gray-Scale Image^۳
Taylor Expansion^۴

برای راحت‌کردن سنجش گوشه می‌توانیم از رابطه زیر بهره ببریم:

$$r = \det(\mathbf{M}) - k \cdot \text{trace}(\mathbf{M}) \quad (3-33)$$

که در آن k پارامتر کنترل حساسیت سنجش گوشه است و معمولاً بین $4/_{\circ}$ و $7/_{\circ}$ انتخاب می‌شود [۴۷]. در رابطه بالا، پارامتر r بزرگ نمایانگر وجود گوشه است. به صورت خلاصه روش الگوریتم شناسایی گوشه به ترتیب زیر است:

۱. محاسبه مشتقات شدت هر ناحیه با استفاده از عملگر سوبل^۱ [۴۸].

$$I_x = G_\sigma^x * I, \quad I_y = G_\sigma^y * I \quad (3-34)$$

۲. محاسبه ضرب مشتقات برای هر ناحیه:

$$I_x^2 = I_x I_x, \quad I_y^2 = I_y I_y, \quad I_{xy} = I_x I_y \quad (3-35)$$

۳. صاف کردن (Smooth) نتایج با کمک یک هسته گوسی^۲:

$$S_x^2 = G_{\sigma^s} * I_x I_x, \quad S_y^2 = G_{\sigma^s} * I_y I_y, \quad S_{xy} = G_{\sigma^s} * I_x I_y \quad (3-36)$$

۴. محاسبه ماتریس \mathbf{M} برای هر ناحیه:

$$\mathbf{M}(x, y) = \begin{bmatrix} S_x^2(x, y) & S_{xy}(x, y) \\ S_{xy}(x, y) & S_y^2(x, y) \end{bmatrix} \quad (3-37)$$

۵. محاسبه پارامتر سنجش گوشه r با استفاده از رابطه ۳-۳

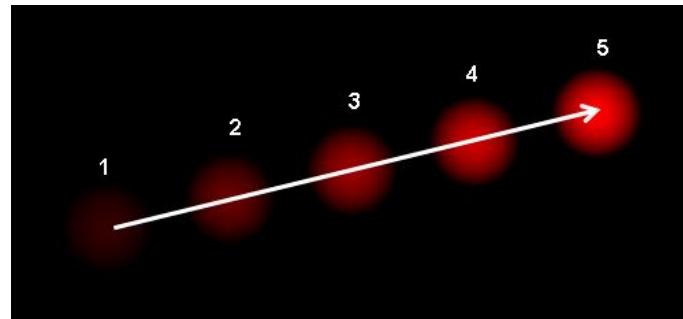
۶. اعمال کردن یک آستانه برای پارامتر r و در نهایت یافتن گوشه‌ها

۳-۴-۲- تشکیل جریان نوری

در بخش قبلی با کمک روش هریس نحوه پیدا کردن نقاط ویژه که گوشه‌های موجود در تصویر هستند، بیان شد. در این بخش در نحوه دنبال کردن و تخمین سرعت این نقاط توسط روش جریان نوری ارایه

Sobel Operation^۱
Gaussian Kernel^۲

خواهد شد. جریان نوری وظیفه تخمین تقریبی حرکت نقطه یا ناحیه‌ای از تصویر در فریم‌های متوالی را نسبت به زمان بر عهده دارد [۴۹] که این موضوع در شکل ۳-۶ به نمایش گذاشته شده است.



شکل ۳-۶ بدار حرکت یک جسم در فریم‌های متوالی [۴۶]

فرض کنید که نقطه‌ای در دنیا واقعی (سه‌بعدی) با $\mathbf{P}_k = [X_k, Y_k, Z_k]^T$ نمایش داده شود، تصویر این نقطه در دوربین به صورت $\mathbf{p}_k = [x_k, x_k]^T$ در می‌آید. همچنین فرض کنید در صورت حرکت نقطه نسبت به دوربین (یا بلعکس)، مسیر طی شده با $\mathbf{p}(t) = [\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n]$ نمایش داده می‌شود که به مجموعه این سرعت‌ها میدان حرکت^۱ می‌گویند. هدف جریان نوری محاسبه این میدان می‌باشد. روش‌های جریان نوری به دو دسته اصلی پراکنده^۲ و متراکم^۳ تقسیم‌بندی کرد. در روش پراکنده میدان حرکت فقط برای تعدادی نقاط خاص سنجیده می‌شود، در حالی که در روش متراکم میدان حرکت برای تمامی نقاط تخمین زده می‌شود. تفاوت بین جریان نوری پراکنده و متراکم را می‌توان در شکل ۷-۳ مشاهده کرد.



شکل ۷-۳ تفاوت بین روش‌های پراکنده و متراکم [۵۰]

Motion Field^۱
Sparse^۲
Dense^۳

در این پژوهش برای تشکیل جریان نوری و تخمین میدان حرکت از یک روش معروف تحت عنوان لوکاس-کنید استفاده می‌شود که مبانی آن در سال ۱۹۸۱ توسط بروس لوکاس^۱ و تاکو کنید^۲ ارایه شد [۵۱]. روش لوکاس-کنید یک روش متراکم بر پایه گرادیان تصویر است. در این روش فرض شده شدت یک نقطه ویژه (همان گوشه‌ها بدست آمده در بخش قبل) در فریم‌های متوالی تغییر نمی‌کند، به عبارت دیگر اگر نقطه ویژه‌ای در زمان t با مختصات x و y در تصویر دو بعدی به اندازه Δx و Δy در فریم بعدی $(t + \Delta t)$ جابه‌جا شود، شدت در هر دو نقطه یکسان فرض می‌شود:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \quad (3-38)$$

اگرچه این فرضیه رابطه (۳۸-۳) به علت تغییرات شرایط نوری و زوایه دید تصویر، یک فرضیه غیرممکن است ولی با توجه به کوچک انتخاب شدن Δx ، Δy و Δt می‌تواند یک تقریب محلی کارآمد باشد (در جابه‌جایی‌های کوچک و محلی این روش کارایی مناسب دارد ولی با طولانی شدن زمان، دچار رانش می‌شود که در پایان این بخش نحوه حل این چالش بیان خواهد شد). همانگونه که در ابتدا گفته شد، هدف از تشکیل جریان نوری محاسبه میدان حرکت (سرعت در راستای x و y) می‌باشد. با کمک بسط تیلور و با فرض کوچک بودن Δx ، Δy و Δt بخش سمت راست رابطه (۳۸-۳) به شکل زیر در می‌آید:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + I_x \Delta x + I_y \Delta y + I_t \Delta t \quad (3-39)$$

که مجدد باتوجه به تساوی رابطه (۳۸-۳) معادله دیفرانسیل زیر بدست می‌آید:

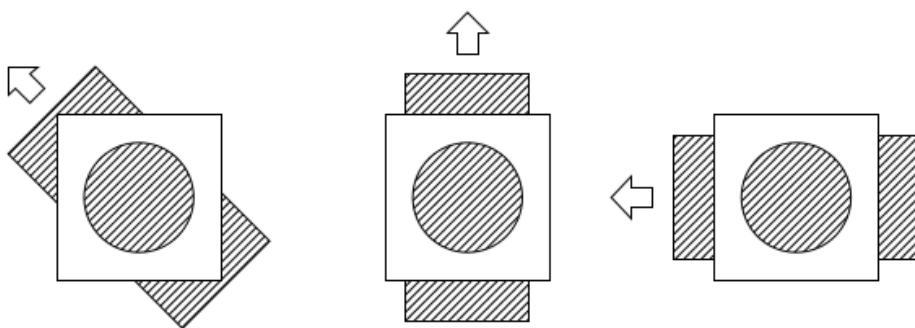
$$I_x \Delta x + I_y \Delta y + I_t \Delta t = 0$$

$$I_x \frac{\Delta x}{\Delta t} + I_y \frac{\Delta y}{\Delta t} + I_t = 0$$

$$I_x v_x + I_y v_y + I_t = 0$$

$$I_x v_x + I_y v_y = -I_t \quad (3-40)$$

در رابطه بالا v_x و v_y سرعت در راستای محور x و y است که دو مجھول معادله دیفرانسیل هستند. در این مسئله در مجھول و یک معادله وجود دارد که معادله را غیرقابل حل می کند که به این چالش، مسئله روزنه^۱ می گویند. مشکل روزنه به این صورت است که اطلاعات موجود در همسایگی نقطه ویژه، شرایط لازم برای محاسبه جهت بردار سرعت را فراهم نمی کند. این مشکل در شکل ۸-۳ به نمایش گذاشته شده است. در این شکل یک روزنه بر روی یک صفحه با الگوی خطوط متناوب قرار گرفته است، صفحه در هر جهتی حرکت کند، از دید روزنه نمی توان جهت را تشخیص داد.



شکل ۸-۳ نمایشی از مسئله روزنه [۵۰]

برای حل این مشکل باید یک معادله جدید اضافه شود تا معادله دیفرانسیل به صورت دو معادله و دو مجھول در بیاید. در روش لوکاس-کنید [۵۱] یک پنجره کوچک به مرکزیت پیکسل مربوط به نقطه ویژه در نظر گرفته می شود که سرعت در این پنجره کوچک و ثابت است. در این روش از یک روش کمینه‌سازی مربعی وزنی^۲ [۵۲] بهره گرفته شده است که وظیفه آن حداقل سازی اختلاف ثابت‌ها می باشد:

$$\text{minimize } \sum_{x,y \in \text{window}} w(x,y) [\nabla I(x,y,t) \cdot \mathbf{v} + I_t(x,y,t)]^2 \quad (3-41)$$

در رابطه بالا کمینه‌سازی قرار است در یک همسایگی از نقطه ویژه \mathbf{p} به مختصات x و y انجام شود. همچنین تابع $w(x,y)$ برای افزایش وزن نقاط نزدیکتر به مرکز در محاسبات تعریف شده است. بنابر مرجع [۵۱] برای هر پیکسل اطراف نقطه ویژه $\mathbf{p}_i = (x_i, y_i) \in \text{window}$ می توان جریان نوری را به شکل زیر تعریف کرد:

Aperture Problem^۱
Weighted Least Squares^۲

$$\mathbf{Av} = \mathbf{b} \rightarrow \mathbf{A} = \begin{bmatrix} \nabla I(x_1, y_1) \\ \nabla I(x_2, y_2) \\ \vdots \\ \nabla I(x_n, y_n) \end{bmatrix}, \mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}, \mathbf{b} = \begin{bmatrix} -I_t(x_1, y_1) \\ -I_t(x_2, y_2) \\ \vdots \\ -I_t(x_n, y_n) \end{bmatrix} \quad (3-42)$$

پس از اعمال رابطه (3-41) سرعت به ترتیب زیر حاصل می‌شود [۵۱]:

$$\mathbf{v} = \mathbf{AB}$$

$$\mathbf{A} = \begin{bmatrix} \sum_i w_i I_x^2(x_i, y_i) & \sum_i w_i I_x(x_i, y_i) I_y(x_i, y_i) \\ \sum_i w_i I_x(x_i, y_i) I_y(x_i, y_i) & \sum_i w_i I_y^2(x_i, y_i) \end{bmatrix} \quad (3-43)$$

$$\mathbf{B} = \begin{bmatrix} -\sum_i w_i I_x(x_i, y_i) I_t(x_i, y_i) \\ -\sum_i w_i I_y(x_i, y_i) I_t(x_i, y_i) \end{bmatrix}$$

اکنون که روش جریان نوری لوکاس-کنید ارایه شد می‌توان به حل مشکل رانش مربوط به دنبال کردن نقطه ویژه در بازه زمانی طولانی پرداخت. همان‌گونه که قبل‌تر ذکر شد در این روش فرض می‌شود که شدت پیکسل مربوط به نقطه ویژه در فریم‌های متوالی ثابت باشد که یک فرض ناممکن است، چراکه با گذر زمان زاویه دید و شرایط نورپردازی (در نتیجه شدت پیکسل‌های تصویر تغییر می‌کند. در نتیجه تغییر شدت پیکسل بعد از گذر زمان طولانی، جریان نوری با خط‌اشکیل می‌شود، برای حل این مشکل نیاز است ظاهر مربوط به نقطه ویژه را در هر فریم جدید بروزرسانی کنیم. برای این کار نیاز به دو جزء داریم:

- جزء توصیف کننده: این بخش شامل یک مربع کوچک است که مرکز آن به نقطه ویژه وصله^۱ می‌شود. در اولین فریم از تصویر این وصله به عنوان وصله^۲ مرجع و در فریم بعدی وصله^۳ نامیده می‌شود.

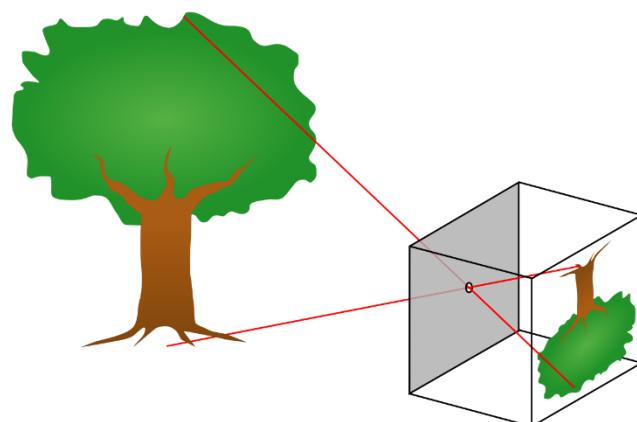
Patch^۱
Reference Patch^۲
Associated Patch^۳

- جزء مقایسه کننده: این جزء وظیفه مقایسه وصله‌های وابسته را با وصله مرجع برعهده دارد.
- برای این قسمت معیار شباهت استفاده شود. برای این معیار از روش میان همبستگی نرمال شده^۱ استفاده می‌شود [۵۴].

برای حل مشکل مربوط به فرض ثابت بودن شدت پیکسل با استفاده از دو جزء بالا، مشخصات نقطه ویژه در هر فریم بروزرسانی می‌شود. بدین ترتیب دیگر با گذر زمان جریان نوری دچار خطای محاسباتی نمی‌شود.

۳-۴-۳- مدل سازی دوربین سوراخ سوزنی

در بخش‌های قبلی نحوه پیدا و دنبال کردن نقاط ویژه بیان شد. با داشتن موقعیت لحظه‌ای نقاط ویژه در تصویر دوبعدی می‌توان عمل مثلث‌بندی را انجام دهیم. قبل بیان روش مثلث‌بندی نیاز است تا مقدماتی در مورد ساختار و نحوه مدل سازی دوربین بیان شود. در این قسمت نحوه مدل سازی دوربین سوراخ سوزنی (شکل ۹-۳) بیان خواهد شد. دوربین سوراخ سوزنی ساده‌ترین مدل از دوربین می‌باشد که عدسی ندارد و معمولاً در اکثر کاربردهای بینایی ماشین، برای پیاده‌سازی معادلات از آن استفاده می‌شود.



شکل ۹-۳ مدل مربوط به دوربین سوراخ سوزنی [۵۵]

^۱ Normalized Cross-Correlation

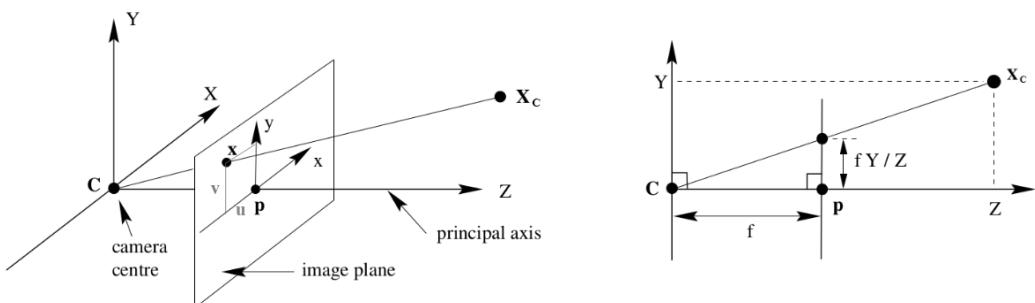
دوربین سوراخ سوزنی (و البته تمام دوربین‌های دیگر) یک نقطه در فضای سه‌بعدی که نسبت به دستگاه مرجع جهانی با $\mathbf{X}^G = [X, Y, Z]$ تعریف می‌شود را در داخل تصویر دو بعدی، به نقطه (یا پیکسل) $\mathbf{x}^{Camera} = [x, y]$ (نسبت به دستگاه مختصات دوربین) تصویر می‌کند. با فرض اینکه ماتریس تبدیل این تصویر با \mathbf{P} نمایش داده شود، رابطه زیر بین \mathbf{X}^G و \mathbf{x}^{Camera} برقرار است:

$$\mathbf{x}^{Camera} = \mathbf{P} \mathbf{X}^G \quad (3-44)$$

ماتریس \mathbf{P} را می‌توان به دو بخش تقسیم کرد و بنابر مرجع [۵۵] رابطه (۴۴-۳) را به صورت زیر درآورد:

$$\mathbf{x}^{Camera} = \mathbf{K} [\mathbf{R} \mid \mathbf{t}] \mathbf{X}^G \quad (3-45)$$

که در رابطه بالا \mathbf{K} مربوط به کالیبراسیون دوربین^۱ و مشخصات ذاتی دوربین^۲ می‌باشد و بخش $[\mathbf{R} \mid \mathbf{t}]$ مربوط به مشخصات بیرونی دوربین^۳ است که ارتباط بین دوربین و دستگاه مرجع جهانی را برقرار می‌کند. در ادامه توضیحات بیشتری در مورد نحوه بدست آمدن رابطه (۴۵-۳) بیان خواهد شد.



شکل ۱۰-۳ ساختار هندسی دوربین سوراخ سوزنی [۵۶]

شکل ۱۰-۳ ساختار یک دوربین سوراخ سوزنی ایده‌آل را نمایش می‌دهد. مرکز دوربین به عنوان مرجع مختصات دوربین انتخاب شده است که تمام نقاط بایستی براساس این مرجع ارایه شوند. صفحه تصویر یک صفحه موازی با محورهای x و y می‌باشد که تمام نقاط دنیای واقعی (همانند \mathbf{X}_c) باید نسبت به دستگاه مرجع مختصات دوربین، بر روی این صفحه تصویر شوند. فاصله این صفحه تا مرکز دوربین

¹ Camera Calibration
² Camera Intrinsic Parameters
³ Camera Extrinsic Parameters

فاصله کانونی f نامیده می‌شود. نقطه \mathbf{X} تصویر نقطه \mathbf{X}_c بر روی صفحه تصویر می‌باشد. محور Z دوربین محور اصلی^۱ و به نقطه p نزدیک نقطه اصلی^۲ گفته می‌شود. براساس شکل ۱۰-۳ (سمت راست) می‌توان مختصات مربوط به نقطه \mathbf{X} (همان u و v) را با داشتن مختصات \mathbf{X}_c و فاصله کانونی f محاسبه کرد. واحد دو مقدار u و v براساس میلی‌متر می‌باشند، نیاز است تا این دو مقدار در واحد پیکسل اندازه-گیری شوند. بدین منظور می‌توان نوشت:

$$f_x = \frac{f}{w}, \quad f_y = \frac{f}{h} \quad (3-46)$$

که در رابطه بالا w و h ابعاد هر پیکسل دوربین می‌باشد و در صورتی که پیکسل‌های دوربین مربعی باشد

$f_x = f_y$ می‌شود. اکنون با داشتن y و f_x می‌توان رابطه تبدیل نقطه سه‌بعدی به دو بعدی را به

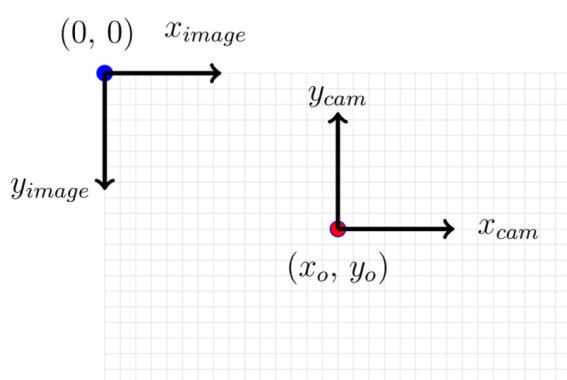
صورت زیر نوشت که در آن دو مقدار u و v براساس واحد پیکسل می‌باشند:

$$u = \frac{Xf_x}{Z}, \quad v = \frac{Yf_y}{Z} \quad (3-47)$$

در روابط بالا دو مقدار u و v براساس دستگاه مرکز دوربین می‌باشند ولی در کاربردهای بینایی ماشین،

مختصات هر نقطه نسبت به دستگاه مرجع تصویر سنجیده می‌شود (شکل ۱۱-۳) که این دستگاه

مرجع در گوشه بالا، سمت چپ تصویر قرار می‌گیرد.



شکل ۱۰-۳ مقایسه دستگاه مختصات تصویر و دوربین [۵۶]

برای توصیف دو مقدار u و v در مختصات مرجع دوربین از روابط زیر استفاده می‌شود:

Principal Axes
Principal Point

$$u = \frac{Xf_x}{Z} + x_0, v = \frac{Yf_y}{Z} + y_0 \quad (3-48)$$

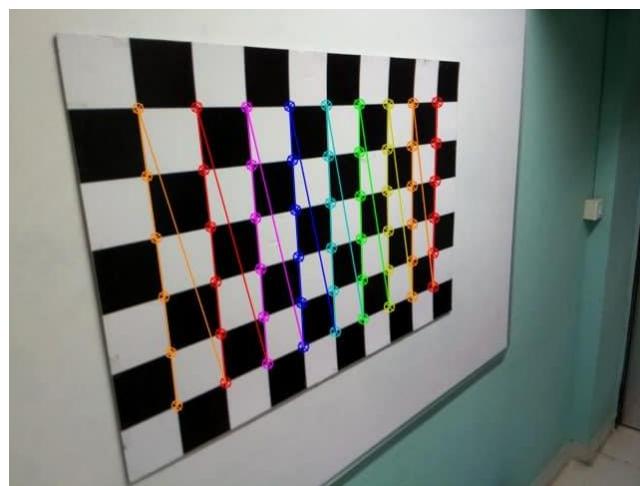
همچنین روابط بالا را می‌توان به شکل ماتریسی زیر درآورد:

$$Z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & x_0 & 0 \\ 0 & f_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (3-49)$$

باتوجه به رابطه (3-49) می‌توان ماتریس کالیبراسیون \mathbf{K} را به شکل زیر تعریف کرد:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3-50)$$

در حوزه بینایی ماشین روش‌های مختلفی برای تعیین ماتریس کالیبراسیون \mathbf{K} ارایه شده است که به صورت خودکار با دریافت تعدادی تصویر این ماتریس را معین می‌کنند، یکی از معروف‌ترین و رایج‌ترین این روش‌ها استفاده از روش صفحه شطرنج می‌باشد [۵۷].



شکل ۱۱-۳ استفاده از صفحه شطرنج برای کالیبراسیون دوربین [۵۷]

رابطه (3-49) نمایانگر تصویر یک نقطه سه‌بعدی در فضای بر روی تصویر دو بعدی می‌باشد. در این رابطه مختصات نقطه سه‌بعدی نسبت به مرکز دوربین در نظر گرفته شده است (شکل ۱۰-۳) ولی مختصات این نقطه نسبت به دستگاه مرجع جهانی مشخص است، بدین منظور از تبدیل زیر جهت تبدیل مختصات استفاده می‌شود:

$$\mathbf{X}^{Camera} = \mathbf{R}(\mathbf{X}^G - \mathbf{C}) \quad (3-51)$$

که در رابطه بالا \mathbf{C} نقطه مرکز دوربین و \mathbf{R} ماتریس دوران دوربین نسبت به مرجع مختصات جهانی می‌باشد. رابطه (۵۱-۳) در فرم ماتریس همگن به شکل زیر در می‌آید:

$$\mathbf{X}^{Camera} = \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X^G \\ Y^G \\ Z^G \\ 1 \end{bmatrix} \quad (3-52)$$

در نهایت با ترکیب روابط (۴۹-۳) و (۵۲-۳) می‌توان به رابطه (۴۵-۳) دست یافت

$$\mathbf{x}^{Camera} = \mathbf{K} [\mathbf{R} \mid \mathbf{t}] \mathbf{X}^G$$

که در آن $\mathbf{t} = -\mathbf{RC}$ می‌باشد.

۳-۴-۴- روشن مثلاً بندی

در بخش‌های قبلی روشن استخراج نقاط ویژه و نحوه دنبال کردن آنها و ساختار و مدل مربوط به دوربین

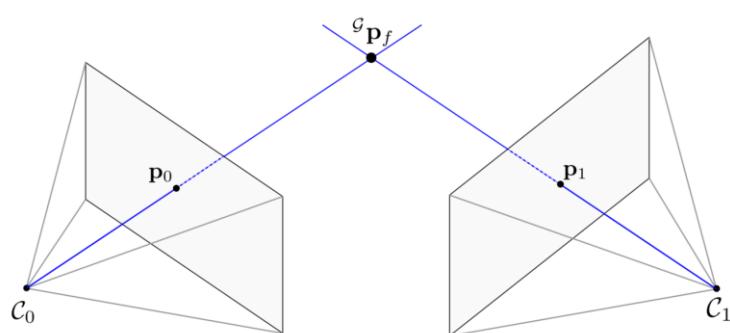
نیز بررسی شد. در این بخش، روشن استخراج موقعیت سه بعدی نقاط ویژه بیان خواهد شد. فرض کنید

که یک نقطه ویژه نمونه در دنیای واقعی در موقعیت p_f^G قرار گرفته است (شکل ۱۲-۳) که توسط

دوربین در موقعیت‌های مختلف C_0, C_1, \dots, C_n بر روی صفحه تصویر در نقاط p_0, p_1, \dots, p_n

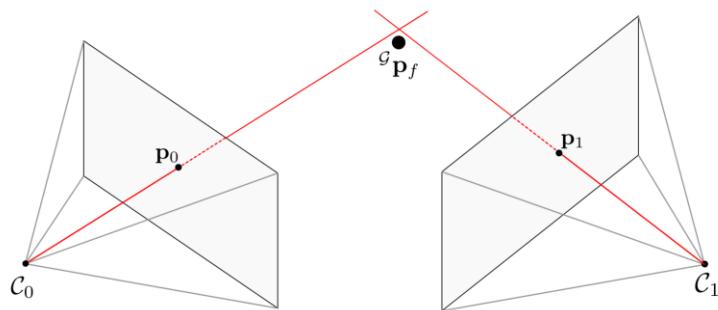
مشاهده می‌شود. در حالت ایده‌آل که دوربین نویز و خطأ نداشته باشد، مشابه شکل ۱۲-۳ خط متصل

کننده مرکز دوربین (C_i) و پیکسل مربوط به نقطه ویژه (p_i) هم‌دیگر را در نقطه p_f^G قطع می‌کنند.



شکل ۱۲-۳ حالت ایده‌آل مثلاً بندی [۵۸]

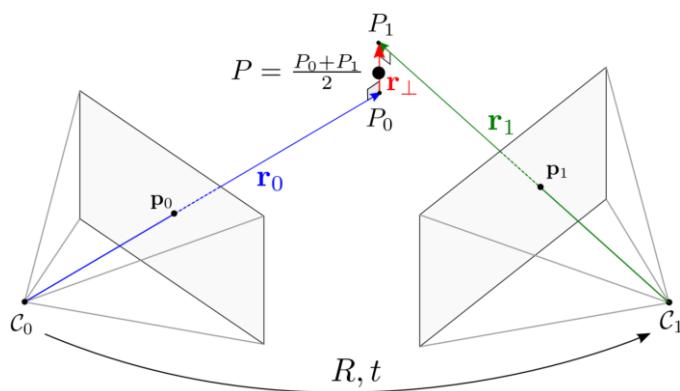
اما در دنیای واقعی این فرضیه امکان پذیر نیست، چراکه دوربین‌ها دارای نویز و خطأ هستند. همچنین داده‌های خروجی دوربین تصاویر پیکسلی هستند (نقطه p_i در مقیاس پیکسل اندازه‌گیری می‌شود) که باعث گستته شدن اندازه‌گیری محل نقطه p_i می‌شود که منجر به خطأ می‌شود (مشابه شکل ۱۳-۳).



شکل ۱۳-۳ تخمین غلط مثلثبندی در حالت عادی [۵۸]

برای حل مشکل خطای مثلثبندی روش‌های مختلفی از جمله تقاطع نقطه میانی^۱ [۵۹] و بهینه‌سازی گوس-نیوتون^۲ [۵۹] ارایه شده است. در روش تقاطع نقطه میانی، همانند شکل ۱۴-۳ سعی می‌شود تا عمود مشترک (r_{\perp}) بین دو خط (r_0 و r_1) پیدا شود و نقطه وسط این خط به عنوان تخمین سه بعدی از نقطه ویژه \mathbf{p} در نظر گرفته شود. فرض کنید \mathbf{R}_i ماتریس دوران دوربین در فریم i باشد. همچنین \mathbf{K} ماتریس کالیبراسیون دوربین (۳-۵۰) و \mathbf{u}_i مختصات دو بعدی پیکسل مربوط به نقطه ویژه باشد، در این صورت رابطه خط به صورت زیر در می‌آید:

$$\mathbf{r}_i = \mathbf{R}_i \mathbf{K}^{-1} \mathbf{u}_i \quad (3-53)$$



Midpoint Intersection Method
Gauss-Newton Minimization

شکل ۱۳-۳ دیاگرام مربوط به روش تقاطع نقطه میانی [۵۸]

می‌توان برای نقطه که به اندازه α_i از مرکز دوربین فاصله دارد، رابطه زیر را نگارش کرد:

$$\mathbf{P}_i = \mathbf{c}_i + \alpha_i \mathbf{r}_i \quad (3-54)$$

برای یافتن عمود مشترک (r_{\perp}) بین دو خط (r_1 و r_0) نیاز است تا α_1 و α_0 را پیدا کنیم که نقطه متناظر

آنها بر روی r_0 و r_1 کمترین فاصله را نسبت به هم داشته باشند:

$$\mathbf{c}_1 + \alpha_1 \mathbf{r}_1 = \mathbf{c}_0 + \alpha_0 \mathbf{r}_0 \quad (3-55)$$

در معادله بالا سه معادله داریم و دو مجھول (α_1 و α_0) که می‌توان با کمک روش حداقل مربعات خطی^۱

[۶۰] این معادله را محاسبه کرد. جواب این دستگاه معادله یک خط می‌باشد که در تقاطع نقطه میانی،

نقطه وسط این خط به عنوان جواب دستگاه معادلات انتخاب می‌شود که همان نقطه P (تخمین از مکان واقعی نقطه ویژه) است. از مزایای اصلی این روش می‌توان به ساده بودن و حجم محاسبات کم آن اشاره کرد، در مقابل دقت پایین (به علت انتخاب نقطه وسط) و عدم کارکرد در صورت موازی بودن r_0 و r_1 جزء معايب آن است. بدین منظور از راه حل دوم، یعنی روش بهینه‌سازی (کمینه‌سازی) گوس-

نیوتون بدین منظور استفاده می‌شود. در این روش یکتابع هزینه به صورت زیر تعریف می‌شود:

$$\mathbf{f}(\boldsymbol{\theta}) = \mathbf{z} - \mathbf{h}(\boldsymbol{\theta}) \quad (3-56)$$

که در آن مقدار مشاهده شده (\mathbf{Z}) و مقدار مورد انتظار ($\mathbf{h}(\boldsymbol{\theta})$) از هم کم می‌شوند. تابع بالا باید

براساس پارامتر $\boldsymbol{\theta}$ در کمینه‌ترین مقدار خود قرار گیرد. در مسئله مثلث‌بندی $\boldsymbol{\theta}$ همان نقطه ویژه در مختصات مرجع جهانی، \mathbf{Z} همان پیکسل مربوط به نقطه ویژه (که از این پس این نقطه ویژه با f_z نمایش داده می‌شود) که برابر $[v_i^j, u_i^j]^T$ می‌باشد. در این مسئله کمینه‌سازی، برای کاهش حجم محاسبات،

از محور r_1 (وبردارهای عمود بر آن) بجای دستگاه مختصات اقلیدسی استفاده می‌شود، بنابراین بردار

خط متصل کننده مرکز دوربین و پیکسل v_z از رابطه زیر محاسبه می‌شود:

$$\mathbf{r}_j^G = \mathbf{p}_{f_j}^G - \mathbf{r}_0^G = \begin{bmatrix} r_j^x \\ r_j^y \\ r_j^z \end{bmatrix} \quad (3-57)$$

با فرض اینکه پارامترهای به صورت زیر تعریف شوند:

$$\alpha_0 = \frac{r_j^x}{r_j^z} \quad \beta_0 = \frac{r_j^y}{r_j^z} \quad \rho_0 = \frac{1}{r_j^z} \quad (3-58)$$

معادله (3-57) به صورت زیر در می‌آید:

$$\mathbf{p}_{f_j}^G = \mathbf{r}_0^G + \frac{1}{\rho_0} \begin{bmatrix} \alpha_0 \\ \beta_0 \\ 1 \end{bmatrix} \quad (3-59)$$

اکنون با کمک رابطه (3-53) می‌توانیم مقدار مورد انتظار ($\mathbf{h}(\boldsymbol{\theta})$) را محاسبه کنیم:

$$\mathbf{h}(\boldsymbol{\theta}) = \mathbf{K} \mathbf{R}_i^T \left(\mathbf{r}_0^G - \mathbf{r}_i^G + \frac{1}{\rho_0} \begin{bmatrix} \alpha_0 \\ \beta_0 \\ 1 \end{bmatrix} \right) \quad (3-60)$$

با کمک رابطه (3-56) و (3-60) می‌توان با کمک یک برنامه کامپیوتروی ساده عمل بهینه‌سازی را انجام داد و نقطه دقیق‌تری برای نقطه ویژه تخمین زد. معمولاً در مثلث‌بندی تخمین اولیه با کمک روش تقاطع نقطه میانی و سایر تخمین‌ها با کمک روش گوس-نیوتون انجام می‌شود.

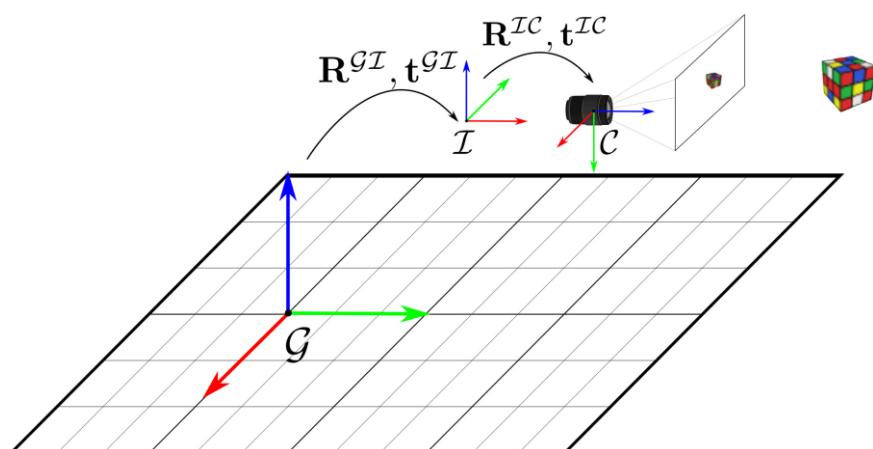
۳-۵- پیاده‌سازی فیلتر کالمن چندحالته مقید

در قسمت‌های قبل در مورد حسگر سنجش اینرسی (IMU)، نحوه ایجاد فیلتر کالمن خطای حالت و همچنین مفاهیم اولیه بینایی ماشین توضیحات مفصلی ارایه شد. در این بخش روش پیاده‌سازی فیلتر کالمن چندحالته مقید (Multi-State Constraint Kalman Filter) یا به اختصار MSCKF، ارایه خواهد شد. این فیلتر در واقع یک روش موقعیت‌سنجی بصری-اینرسی برگرفته از فیلتر کالمن توسعه یافته (EKF) یا به اختصار Extended Kalman Filter می‌باشد. این روش برخلاف روش‌های موقعیت‌سنجی مبتنی بر EKF، از محیط نقشه ایجاد نمی‌کند، بلکه در آن از یک پنجره کشویی برای ذخیره‌سازی موقعیت‌ها قبلی در جهت تخمین موقعیت‌های جدید استفاده می‌شود. در نگاه اول اگرچه ایجاد نقشه (که با ذخیره موقعیت تمام نقاط ویژه فریم‌ها ایجاد می‌شود) کاربردی به نظر می‌رسد ولی

ایجاد نقشه و موقعیت‌سنجی براساس آن علاوه بر پایین آوردن دقت موقعیت‌سنجی، حجم محاسبات را بالا می‌برد و منجر به غیرقابل استفاده شدن الگوریتم در سخت‌افزارهای ارزان قیمت (نظیر کامپیوتر کوچک) می‌شود. در مقابل در روش MSCKF حجم محاسبات پایین می‌باشد و به صورت خطی با تعداد نقاط ویژه افزایش پیدا می‌کند. در این بخش، ابتدا در قسمت ۱-۵-۳ درمورد کلیات الگوریتم روش MSCKF توضیحات ارایه خواهد شد، در قسمت‌های ۲-۵-۳ تا ۴-۵-۳ در بردار حالت، بخش انتشار و تقویت کننده بررسی خواهد شد. در قسمت ۵-۵-۳ در مورد مدل مشاهده‌گر و در بخش پایانی در مورد بخش اصلاح توضیحات ارایه خواهد شد.

۱-۵-۳-۱ بررسی اجمالی

در روش MSCKF هدف تخمین موقعیت و جهت‌گیری حسگر سنجش اینرسی (IMU) با کمک داده‌های دوربین نسبت به دستگاه مرجع جهانی (G) می‌باشد. برای کاهش محاسبات الگوریتم فرض می‌شود که دستگاه مرجع جهانی نقطه‌ای باشد که در آن الگوریتم شروع به کار می‌کند، همچنین محور z حسگر سنجش اینرسی (IMU) در راستای شتاب گرانش تنظیم قرار می‌گیرد. لیست چارچوب‌های هندسی مورد نیاز در این الگوریتم را می‌توانید در شکل ۱۴-۳ مشاهده کنید.



شکل ۱۴-۳ وضعیت هندسی چارچوب‌های مختلف [۴۰]

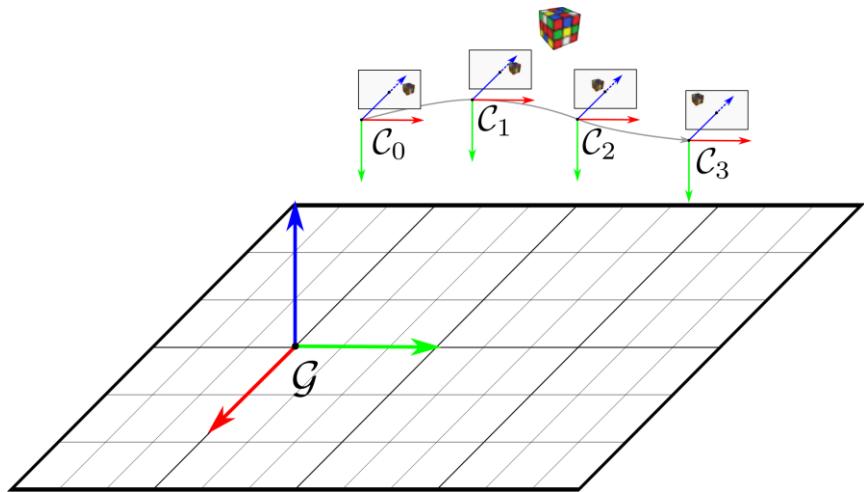
در روش MSCKF به سه چارچوب مختصاتی نیاز است:

- چارچوب مرجع یا دستگاه مختصات جهانی که با G نمایش داده شده و همان‌گونه که ذکر شد، محل شروع الگوریتم انتخاب می‌شود.
- چارچوب چسبیده به حسگر سنجش اینرسی (IMU) که با I نمایش داده می‌شود و با تبدیل $[R^{GI} \mid t^{GI}]$ نسبت به چارچوب مرجع معرفی می‌شود. لازم به ذکر است که داده‌های حسگر سنجش اینرسی (با فرکانسی بین ۱۰۰ تا ۱۲۰ هرتز) از این موقعیت منتشر می‌شوند.
- چارچوب چسبیده به دوربین که با C نمایش داده می‌شود و از تبدیل $[R^{IC} \mid t^{IC}]$ نسبت به چارچوب I معین می‌شود. موقعیت و جهت‌گیری این چارچوب نسبت به چارچوب I از پیش تعیین شده و ثابت می‌باشد. همچنین داده‌های تصویری دوربین (معمولاً با فرکانس ۳۰ هرتز) از چارچوب C منتشر می‌شود.

با استفاده از چارچوب‌های معرفی شده می‌توان الگوریتم موقعیت‌سنجی را پیاده‌سازی کرد. در این الگوریتم ابتدا، هر موقع که داده‌ای جدید از حسگر سنجش اینرسی (IMU) دریافت شود، با کمک روابط ارایه شده در بخش ۳-۳ توسعه حالت اسمی و حالت خطا انتشار داده می‌شود. به عبارت دیگر از شتاب خطی و سرعت زاویه‌ای از حسگر سنجش اینرسی (IMU) دریافت شده و موقعیت، سرعت، جهت‌گیری و همچنین ماتریس کواریانس جدید پس از تخمین (پیش‌بینی) منتشر می‌شود.

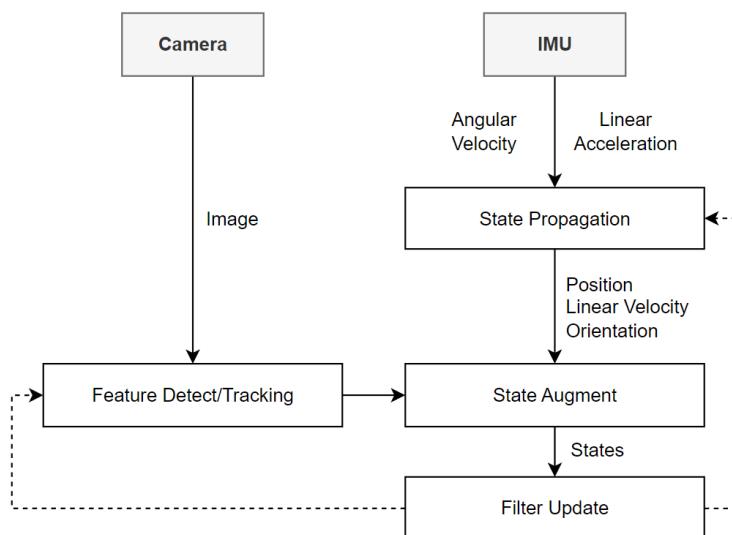
در قدم دوم هنگامی که تصویری جدید توسط دوربین منتشر شود، حالت سیستم با افزودن موقعیت دوربین به پنجره کشویی افزایش می‌یابد^۱، همچنین ماتریس کواریانس نیز بروزرسانی می‌شود. بعد از انجام افزایش حالت^۲ با کمک الگوریتم‌های هریس و لوکاس کنید (که در بخش ۴-۳ بررسی شد) نقاط ویژه برای تصویر جدید مجدد شناسایی و دنبال می‌شوند، توجه شود که این نقاط ویژه (حتی نقاطی که از پنجره دید تصویر خارج شده‌اند) در یک لیست در حافظه برنامه ذخیره می‌شوند که به این لیست پنجره کشویی (شکل ۱۴-۳) می‌گویند. از داده‌های قدیمی‌تر پنجره کشویی برای بروزرسانی فیلتر

استفاده می‌شود. این بروزرسانی با کمک روش مثلثبندی و با کمک داده‌های قدیمی‌تر پنجره انجام می‌شود (که دقیق در تخمین را بسیار افزایش می‌دهد).



شکل ۱۴-۳ نمایش چهار داده آخر پنجره کشویی [۴۰]

درنهایت، خطای بازپرداخت^۱ تخمین موقعیت نقاط ویژه در مقایسه با مقادیر مشاهده شده، محاسبه می‌شود که به آن باقی‌مانده فیلتر می‌گویند. از باقی‌مانده فیلتر برای اصلاح فیلتر استفاده می‌شود. ساختار کلی الگوریتم MSCKF را می‌توانید در شکل ۱۵-۳ مشاهده نمایید.



شکل ۱۵-۳ ساختار کلی الگوریتم MSCKF

^۱ Reprojection Errors

در بخش‌های بعدی توضیحات جامع‌تری در مورد هر کدام از بخش‌های الگوریتم ارایه خواهد شد.

۳-۵-۲- بررسی ساختار بردار حالت

بردار حالت به برداری گفته می‌شود که حاوی پارامترهایی است که باستی دائمًا توسط فیلتر تخمین زده شوند. در این پژوهش فرض می‌شود که بردار حالت اسمی (\mathbf{x}) از دو بخش مجزا تشکیل شده باشد، یک بخش مربوط به متغیرهای حالت حسگر سنجش اینرسی (IMU) می‌باشد که با \mathbf{x}_{imu} نمایش داده می‌شود و به صورت زیر تعریف می‌شود:

$$\mathbf{x}_{imu} = \begin{bmatrix} \mathbf{p}^{GI^T} & \mathbf{v}^{GI^T} & \mathbf{q}^{GI^T} & \mathbf{b}_a^T & \mathbf{b}_\omega^T \end{bmatrix}^T \quad (3-61)$$

و یک بخش مربوط به حالت‌های اضافه شونده پنجره کشویی می‌باشد که به انتهای بردار ارایه شده رابطه (۶۱-۳) اضافه می‌شوند:

$$\mathbf{x} = [\mathbf{x}_{imu} \quad \boldsymbol{\pi}_1^T \quad \boldsymbol{\pi}_2^T \quad \dots \quad \boldsymbol{\pi}_n^T]^T \quad (3-62)$$

در رابطه بالا $\boldsymbol{\pi}_i$ مربوط به موقعیت و جهت‌گیری دوربین در هر فریم i ام می‌باشد که به صورت زیر تعریف می‌شود:

$$\boldsymbol{\pi}_i = \begin{bmatrix} \mathbf{p}^{GC_i^T} & \mathbf{q}^{GC_i^T} \end{bmatrix}^T \quad (3-63)$$

لازم به ذکر است که داده‌های مربوط به حسگر سنجش اینرسی (IMU) در چارچوب I و داده‌های مربوط به دوربین در چارچوب C سنجیده می‌شوند. اکنون که بردار حالت اسمی مشخص شد، می‌توان با توجه به روابطی که در بخش ۳-۳ ارایه شده، به طریق مشابه بردار حالت خطای محاسبه کرد، بردار حالت خطای برای حسگر سنجش اینرسی (IMU) به ترتیب زیر بدست می‌آید:

$$\delta \mathbf{x}_{imu} = \begin{bmatrix} \delta \mathbf{p}^{GI^T} & \delta \mathbf{v}^{GI^T} & \delta \boldsymbol{\theta}^{GI^T} & \delta \mathbf{b}_a^T & \delta \mathbf{b}_\omega^T \end{bmatrix}^T \quad (3-64)$$

و به طریق مشابه با رابطه (۶۲-۳) بردار حالت خطای کل به ترتیب زیر معین می‌شود:

$$\delta \mathbf{x} = [\delta \mathbf{x}_{imu} \quad \delta \boldsymbol{\pi}_1^T \quad \delta \boldsymbol{\pi}_2^T \quad \dots \quad \delta \boldsymbol{\pi}_n^T]^T \quad (3-65)$$

که در آن خطای موقعیت و جهت‌گیری دوربین در فریم شماره i می‌باشد:

$$\delta \boldsymbol{\pi}_i = [\delta \mathbf{p}^{GC_i T} \quad \delta \boldsymbol{\theta}^{GC_i T}]^T \quad (3-66)$$

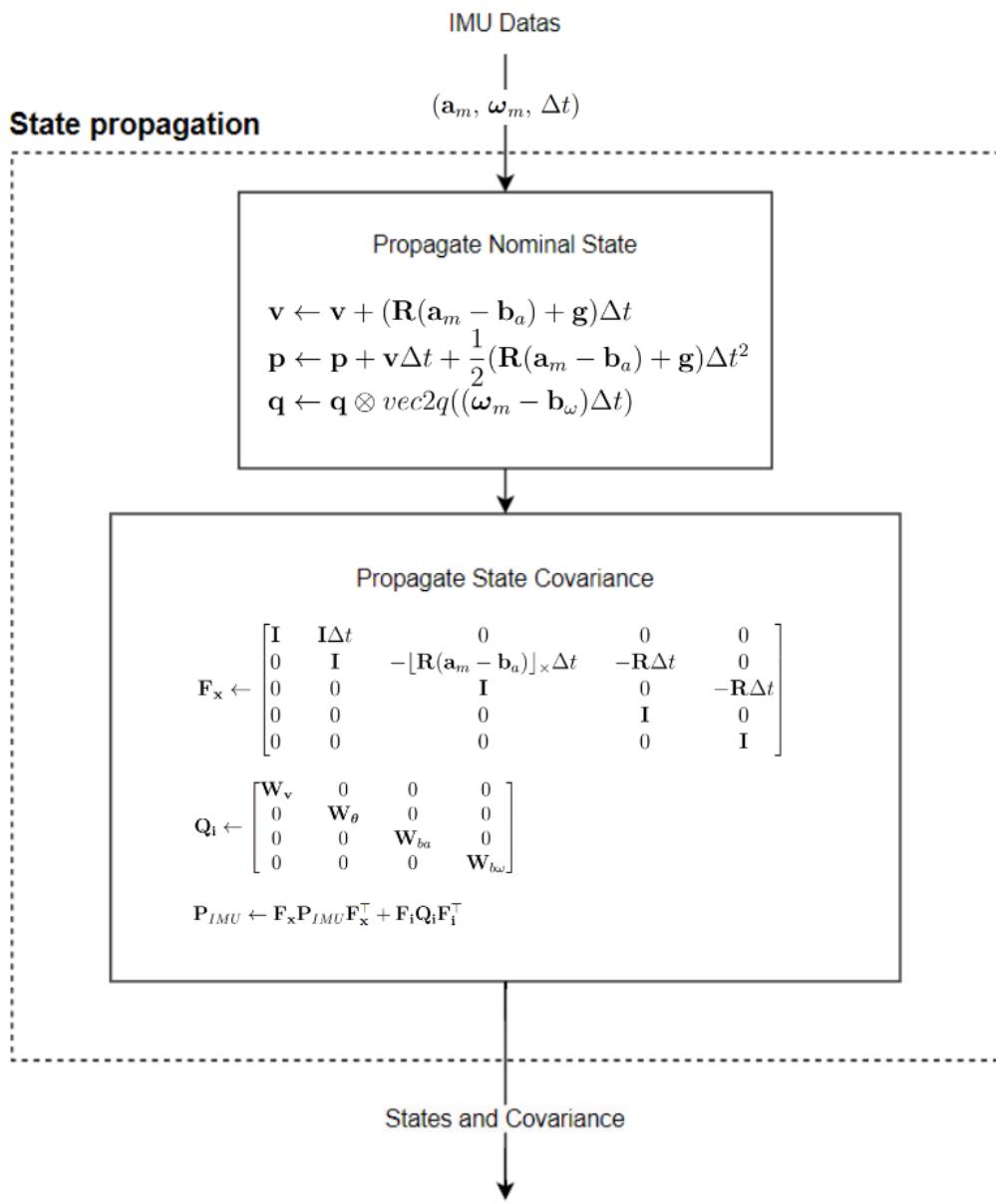
همچنین ماتریس کواریانس که نمایانگر عدم قطعیت مربوط به حالت خطا است به شکل زیر می‌باشد:

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{II} & \mathbf{P}_{IW} \\ -\mathbf{P}_{IW} & \mathbf{P}_{WW} \end{bmatrix} \quad (3-67)$$

که در آن \mathbf{P}_{II} کواریانس مربوط به حالت خطا حسگر سنجش اینرسی، \mathbf{P}_{WW} کواریانس مربوط به حالت خطای پارامترهای دوربین و \mathbf{P}_{IW} کواریانس مربوط به خطای ترکیب دوربین و حسگر سنجش اینرسی است. (IMU)

۳-۵-۳- توسعه بخش انتشار حالت

بخش انتشار حالت هر بار که حسگر سنجش اینرسی (IMU) یک داده جدید اندازه‌گیری و ارسال کند، اجرا می‌شود. این بخش وظیفه دارد تا با کمک داده‌های دریافتی، بردار حالت و ماتریس کواریانس را تخمین بزند و آنرا برای مرحله بعد منتشر کند. در این مرحله از روابط ارایه شده در بخش ۳-۳ برای توسعه الگوریتم استفاده می‌شود. الگوریتم مربوط به این مرحله در شکل ۱۶-۳ قابل مشاهده است. توجه شود که الگوریتم ارایه شده، یکتابع است که در ورودی داده‌های حسگر سنجش اینرسی (IMU) را دریافت و در خروجی متغیرهای حالت و ماتریس کواریانس مربوط به آن حسگر را منتشر می‌کند.



شکل ۱۶-۳ ساختار کلی بخش انتشار حالت

۴-۵-۳- بخش افزودن حالت

هرگاه که یک داده‌ی تصویری جدید توسط دوربین انتشار داده شود، بایستی موقعیت دوربین در لحظه ثبت تصویر به بردار حالت اضافه شود. این بخش این عمل را انجام می‌دهد، بدین منظور بخش افزودن حالت نامیده می‌شود. در این بخش دو مورد بایستی بروزرسانی شود:

• بردار حالت

برای بروزرسانی بردار حالت صرفا نیاز است تا موقعیت جدید دوربین محاسبه شود و به انتهای بردار حالت اضافه شود. به این منظور لازم است تا موقعیت و جهت‌گیری حسگر سنجش اینرسی (IMU) که در بخش انتشار حالت محاسبه شده‌اند، با یک تبدیل به چارچوب دوربین انتقال داده شوند (مشابه شکل ۱۴-۳). بدین منظور در لحظه N می‌توان نوشت:

$$\mathbf{p}^{GC_N} = \mathbf{p}^{GI_N} + \mathbf{R}^{GI} \mathbf{p}^{IC} \quad (3-68-1)$$

$$\mathbf{q}^{GC_N} = \mathbf{q}^{GI_N} \otimes \mathbf{q}^{IC} \quad (3-68-2)$$

که در رابطه بالا، \mathbf{p}^{IC} و \mathbf{q}^{IC} ماتریس‌های انتقال از حسگر سنجش اینرسی (IMU) به دوربین می‌باشد که این ماتریس‌ها در طول فرآیند موقعیت‌سنجی ثابت می‌باشند. پس محاسبه موقعیت و جهت‌گیری (بردار کواترنیون) به شکل زیر در داخل بردار قرار گرفته و به پنجره کشویی حالت اسمی اضافه می‌شود:

$$\boldsymbol{\pi}_N = \begin{bmatrix} \mathbf{p}^{GC_N T} & \delta \boldsymbol{\theta}^{GC_N T} \end{bmatrix}^T \quad (3-69)$$

اکنون نوبت به بروزرسانی ماتریس کواریانس می‌رسد. این ماتریس زمانی که عضو جدیدی به پنجره کشویی (یا همان بردار حالت) اضافه می‌شود، باید بروزرسانی شود تا عدم قطعیت‌های مربوط به آن موقعیت لحاظ شود. لازم به ذکر است که در الگوریتم MSCKF با اضافه شدن یک عضو جدید به بردار حالت، تمام درایه‌های ماتریس کواریانس تغییر نمی‌کنند بلکه فقط درایه‌های متناظر با عضو جدید بردار حالت دچار تغییر می‌شوند.

باتوجه به اینکه ماتریس کواریانس برای تخمین حالت خطای استفاده می‌شود، پس مقدار آن به ژاکوبین خطای موقعیت ($\delta \mathbf{p}^{GC_N}$) جدید بستگی دارد. بدین منظور، با درنظر گرفتن حالت خطای ژاکوبین تبدیل (۳-۶۸) محاسبه می‌شود. توجه شود که فقط درایه‌های مرتبط با موقعیت و جهت‌گیری دچار تغییر می‌شوند:

$$J_{\boldsymbol{\pi}_N} = \frac{\partial \delta \boldsymbol{\pi}_N}{\partial \delta \mathbf{x}} = \quad (3-70)$$

$$\begin{bmatrix} \frac{\partial \delta \mathbf{p}^{GC_N}}{\partial \delta \mathbf{p}^{GI}} & \mathbf{0} & \frac{\partial \delta \mathbf{p}^{GC_N}}{\partial \delta \boldsymbol{\theta}^{GI}} & \cdots & \mathbf{0} \\ \frac{\partial \delta \boldsymbol{\theta}^{GC_N}}{\partial \delta \mathbf{p}^{GI}} & \mathbf{0} & \frac{\partial \delta \boldsymbol{\theta}^{GC_N}}{\partial \delta \boldsymbol{\theta}^{GI}} & \cdots & \mathbf{0} \end{bmatrix}$$

برای محاسبه ماتریس ژاکوبین از رابطه (۳-۶۸-۱) استفاده می‌شود که می‌توان آنرا با کمک مجموعه

روابط ۲۸-۳ به صورت زیر نوشت:

$$\mathbf{p}^{GC_N} + \delta \mathbf{p}^{GC_N} = \mathbf{p}^{GI} + \delta \mathbf{p}^{GI} + (\mathbf{I} + [\delta \boldsymbol{\theta}^{GI}]_{\times}) \mathbf{R}^{GI} \mathbf{p}^{IC}$$

که در آن \mathbf{p}^{IC} یک مقدار ثابت و از پیش تعریف شده می‌باشد:

$$\mathbf{p}^{GC_N} + \delta \mathbf{p}^{GC_N} = \mathbf{p}^{GI} + \delta \mathbf{p}^{GI} + (\mathbf{R}^{GI} + [\delta \boldsymbol{\theta}^{GI}]_{\times} \mathbf{R}^{GI}) \mathbf{p}^{IC}$$

$$\mathbf{p}^{GC_N} + \delta \mathbf{p}^{GC_N} = \mathbf{p}^{GI} + \delta \mathbf{p}^{GI} + \mathbf{R}^{GI} \mathbf{p}^{IC} + [\delta \boldsymbol{\theta}^{GI}]_{\times} \mathbf{R}^{GI} \mathbf{p}^{IC}$$

با ایزوله کردن اجزاء حالت خط رابطه زیر حاصل می‌شود:

$$\delta \mathbf{p}^{GC_N} = \delta \mathbf{p}^{GI} + [\delta \boldsymbol{\theta}^{GI}]_{\times} \mathbf{R}^{GI} \mathbf{p}^{IC}$$

با توجه به اینکه $[\mathbf{v}]_{\times} \mathbf{w} = -[\mathbf{w}]_{\times} \mathbf{v}$ می‌باشد:

$$\delta \mathbf{p}^{GC_N} = \delta \mathbf{p}^{GI} - [\mathbf{R}^{GI} \mathbf{p}^{IC}]_{\times} \delta \boldsymbol{\theta}^{GI}$$

در نهایت معادله مشتقات جزئی به شکل زیر حاصل می‌شود:

$$\frac{\partial \delta \mathbf{p}^{GC_N}}{\partial \delta \mathbf{p}^{GI}} = \mathbf{I} \quad \frac{\partial \delta \mathbf{p}^{GC_N}}{\partial \delta \boldsymbol{\theta}^{GI}} = -[\mathbf{R}^{GI} \mathbf{p}^{IC}]_{\times} \quad (3-71)$$

به همین ترتیب، با توسعه رابطه (۳-۶۸-۲) می‌توان برای جهتگیری (کواتریون‌ها) نیز مشتقات جزئی

را محاسبه کرد. حاصل نهایی به ترتیب زیر می‌باشد:

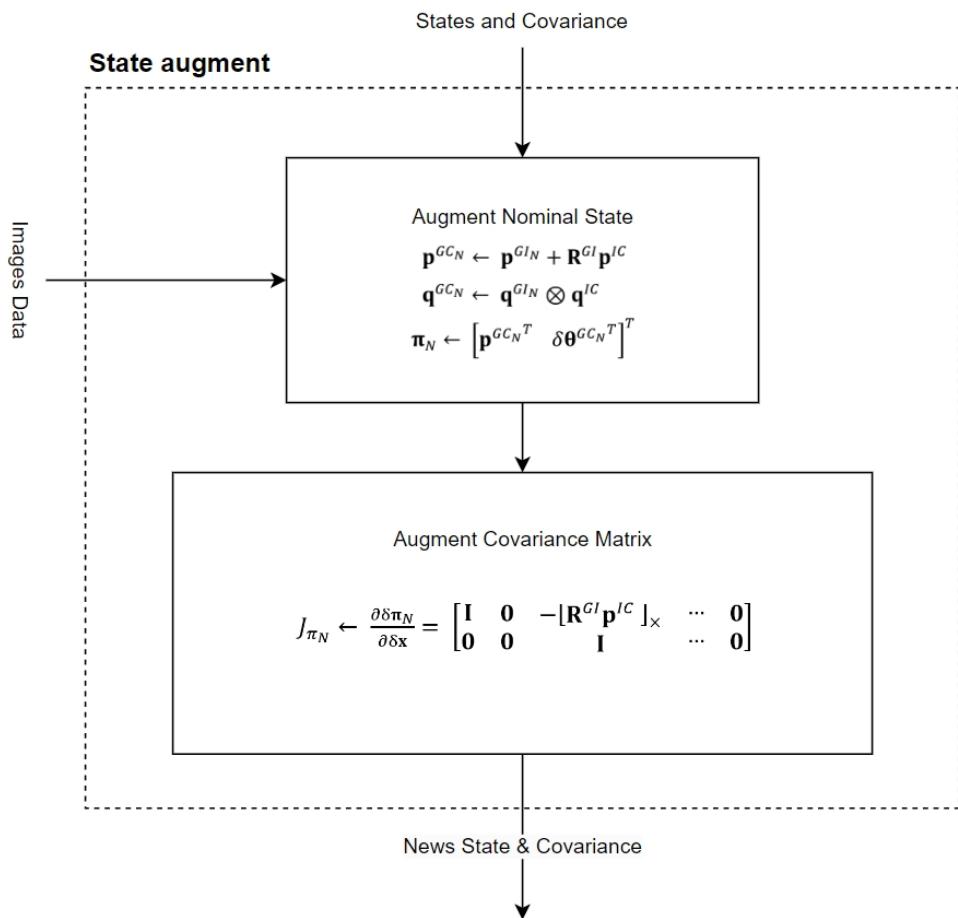
$$\frac{\partial \delta \boldsymbol{\theta}^{GC_N}}{\partial \delta \mathbf{p}^{GI}} = \mathbf{0} \quad \frac{\partial \delta \boldsymbol{\theta}^{GC_N}}{\partial \delta \boldsymbol{\theta}^{GI}} = \mathbf{I} \quad (3-72)$$

اکنون با استفاده از رابطه‌های (۳-۷۱-۱) و (۳-۷۲-۱) می‌توان ماتریس ژاکوبین (۳-۷۰-۱) را به صورت زیر

تکمیل کرد:

$$J_{\pi_N} = \frac{\partial \delta \boldsymbol{\pi}_N}{\partial \delta \mathbf{x}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & -[\mathbf{R}^{GI} \mathbf{p}^{IC}]_{\times} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{I} & \cdots & \mathbf{0} \end{bmatrix} \quad (3-70)$$

که ماتریس ژاکوبین بالا فقط به درایه‌های متناظر با حالت جدید در ماتریس کواریانس اعمال می‌شود.
ساختمان کلی بخش افزودن حالت (که به صورت تابع است) را می‌توانید در دیاگرام موجود در شکل ۱۷ مشاهده نمایید.



شکل ۱۷-۳ ساختار کلی بخش افزایش حالت

۱-۵-۳-۳- توسعه مدل رؤیت^۱

همان‌گونه که در بخش‌های قبلی مطرح شد، در مرحله انتشار حالت مربوط به حسگر سنجش اینرسی (IMU)، خطأ با زمان انباشته می‌شود. در این بخش، روش اصلاح این خطأ با کمک دوربین ارایه می‌شود که به این منظور، از یک مدل رؤیت استفاده خواهد شد. در مدل رؤیت، ابتدا نیاز است تا با کمک مفاهیمی که در بخش ۴-۳ ارایه شد، نقاط ویژه (که گوشه‌های موجود در تصویر هستند) استخراج

¹ Observation Model

شوند. البته می‌توان جزئیات دیگر موجود در تصویر همانند لبه‌ها یا بافت‌های خاص دیگر را به عنوان نقاط ویژه انتخاب کرد ولی در این پژوهش، از گوشی‌ها (به علت سهولت در تشخیص و دنبال کردن) به عنوان نقاط ویژه استفاده خواهد شد. قبل از شروع توسعه مدل رؤیت، نیاز است تا مفروضات لازم بیان شود، فرض کنید که نقاط ویژه با \mathbf{z}_j نمایش داده شود که در آن \mathbf{z} -شمارشگر نقاط ویژه است. همچنین مقادیر رؤیت شده با \mathbf{z}_i^j نمایش داده شود که حاوی موقعیت نقطه ویژه \mathbf{z} -ام در تصویر فریم i -ام است (با داده می‌شود).

در قدم اول توسعه مدل رویت، با استفاده از روش مثلث‌بندی گوس-نیوتون (بخش ۴-۳) موقعیت نقطه ویژه مشخص می‌شود که با $\widehat{\mathbf{p}}_{f_j}^G$ نمایش داده می‌شود. نیاز است تا این نقطه در چارچوب چسبیده به دوربین سنجیده شود، بدین منظور از تبدیل زیر استفاده می‌شود:

$$\widehat{\mathbf{p}}_{f_j}^{C_i} = \mathbf{R}^{GCiT} \left(\widehat{\mathbf{p}}_{f_j}^G - \mathbf{p}^{GC_i} \right) \quad (3-71)$$

همچنین با توجه رابطه (۴۸-۳) برای $\widehat{\mathbf{z}}_i^j$ می‌توانیم بنویسیم:

$$\widehat{u}_j^i = \frac{x_j^i f_x}{z_j^i} + x_0 \quad \widehat{v}_j^i = \frac{y_j^i f_y}{z_j^i} + y_0 \quad (3-72)$$

که در آن x_0 ، y_0 و f_x ، f_y پارامترهای ذاتی دوربین می‌باشد (در بخش ۴-۳ ارایه شده است). اکنون می‌توان پارامتر باقی‌مانده فیلتر^۱ را براساس مرجع [۳۸] به صورت زیر نگارش کرد:

$$\mathbf{r}_i^j = \mathbf{z}_i^j - \widehat{\mathbf{z}}_i^j \quad (3-73)$$

با جایگذاری و خطی‌سازی، رابطه (۷۳-۳) به صورت زیر حاصل می‌شود:

$$\mathbf{r}_i^j = \mathbf{H}_{\delta x_i}^j \delta x + \mathbf{H}_{f_j}^j \delta \mathbf{p}_{f_j} + \mathbf{n} \quad (3-74)$$

در رابطه بالا ماتریس $\mathbf{H}_{\delta x}$ ژاکوبین اندازه‌گیری نسبت به حالت خطا (مربوط به حسگر IMU) می‌باشد و ماتریس \mathbf{H}_{f_j} ژاکوبین اندازه‌گیری خطای موقعیت نقاط ویژه (مربوط به دوربین) است. همچنین \mathbf{n} به

Filter Residual¹

عنوان نویز تصویر در نظر گرفته می‌شود که معمولاً میانگین آن نزدیک به صفر است و به حالت خطابه (۳) غیر وابسته است. و ماتریس $\mathbf{H}_{\delta x}$ با کمک قاعده مشتق زنجیره‌ای به شکل زیر قابل محاسبه است:

$$\mathbf{H}_{\delta x_i}^j = \frac{\partial \mathbf{z}_i^j}{\partial \delta x} = \frac{\partial \mathbf{z}_i^j}{\partial \mathbf{p}_{f_j}^{c_i}} \cdot \frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \delta x}$$

که در آن بخش $\frac{\partial \mathbf{z}_i^j}{\partial \mathbf{p}_{f_j}^{c_i}}$ از رابطه (۷۲-۳) قابل محاسبه است:

$$\mathbf{H}_{z_i}^j = \frac{\partial \mathbf{z}_i^j}{\partial \mathbf{p}_{f_j}^{c_i}} = \begin{bmatrix} \frac{f_x}{z_i^j} & 0 & -\frac{f_x \cdot x_i^j}{(z_i^j)^2} \\ 0 & \frac{f_y}{z_i^j} & -\frac{f_x \cdot y_i^j}{(z_i^j)^2} \end{bmatrix} \quad (۳-۷۵)$$

همچنین بخش $\frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \delta x}$ با جایگذاری در رابطه (۷۱-۳) به شکل زیر در می‌آید:

$$\frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \delta \mathbf{p}^{GC_i}} = -\mathbf{R}^{GC^T} \quad \frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \delta \theta^{GC_i}} = -\mathbf{R}^{GC^T} [(\mathbf{p}_{c_i}^G - \mathbf{p}_{f_i}^G)]_x$$

که دو مقدار بالا را می‌توان در ماتریس ذخیره کرد:

$$\mathbf{H}_{\pi_i}^j = \frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \pi_i} = [-\mathbf{R}^{GC^T} \quad -\mathbf{R}^{GC^T} [(\mathbf{p}_{c_i}^G - \mathbf{p}_{f_i}^G)]_x]$$

در نهایت ماتریس $\mathbf{H}_{\delta x_i}^j$ به صورت زیر در می‌آید:

$$\mathbf{H}_{\delta x_i}^j = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \cdots & -\mathbf{H}_{z_i}^j \mathbf{R}^{GC^T} & -\mathbf{H}_{z_i}^j \mathbf{R}^{GC^T} [(\mathbf{p}_{c_i}^G - \mathbf{p}_{f_i}^G)]_x & \cdots \end{bmatrix} \quad (۳-۷۶)$$

برای محاسبه ماتریس \mathbf{H}_{f_j} نیز به طریق مشابه از قاعده مشتق زنجیره‌ای بهره می‌بریم:

$$\mathbf{H}_{f_i}^j = \frac{\partial \mathbf{z}_i^j}{\partial \mathbf{p}_{f_j}^G} = \frac{\partial \mathbf{z}_i^j}{\partial \mathbf{p}_{f_j}^{c_i}} \cdot \frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \mathbf{p}_{f_j}^G}$$

قسمت اول یعنی $\frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \mathbf{p}_{f_j}^G}$ همان رابطه (۷۶-۳) می‌باشد و قسمت $\frac{\partial \mathbf{z}_i^j}{\partial \mathbf{p}_{f_j}^{c_i}}$ از رابطه (۷۱-۳) محاسبه می‌شود.

شود:

$$\frac{\partial \mathbf{p}_{f_j}^{c_i}}{\partial \mathbf{p}_{f_j}^G} = [\mathbf{R}_{c_i}^G]^t$$

در نهایت $\mathbf{H}_{f_j}^j$ از رابطه زیر محاسبه می‌شود:

$$\mathbf{H}_{f_j}^j = \mathbf{H}_{z_i}^j [\mathbf{R}_{c_i}^G]^t \quad (3-77)$$

اکنون با داشتن $\mathbf{H}_{f_j}^j$ ها و $\mathbf{H}_{\delta x_i}^j$ که صرفاً مربوط به یک رؤیت می‌باشند، می‌توان ماتریس‌های ژاکوپین کل (حاوی تمام رؤیت‌ها) را به صورت زیر نگارش کرد. ماتریس ژاکوپین مربوط به حالت خطابه:

صورت زیر است:

$$\mathbf{H}_{\delta x}^j = \begin{bmatrix} 0 & \mathbf{H}_{\pi_0}^j & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{H}_{\pi_1}^j & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{H}_{\pi_M}^j \end{bmatrix} \quad (3-78)$$

همنچنین ماتریس ژاکوپین مربوط به خطاب نقاط ویژه نیز به ترتیب زیر است:

$$\mathbf{H}_f^j = \begin{bmatrix} \mathbf{H}_{f_0}^j \\ \mathbf{H}_{f_1}^j \\ \vdots \\ \mathbf{H}_{f_M}^j \end{bmatrix} \quad (3-79)$$

رابطه (۷۴-۳) مجدد به صورت زیر بازنویسی می‌شود:

$$\mathbf{r}^j = \mathbf{H}_{\delta x}^j \delta x + \mathbf{H}_f^j \delta \mathbf{p}_{f_j} + \mathbf{n} \quad (3-80)$$

که در آن \mathbf{r}^j به صورت زیر نگارش می‌شود:

$$\mathbf{r}^j = \begin{bmatrix} \mathbf{r}_0^j \\ \mathbf{r}_1^j \\ \vdots \\ \mathbf{r}_M^j \end{bmatrix} \quad (3-81)$$

برای اینکه بتوان از رابطه (۳-۸۰) در قسمت بعدی (بروزرسان فیلتر) استفاده کرد، نیاز است تا آنرا در

فرمت زیر بیان کنیم:

$$\mathbf{r} = \mathbf{H}\delta x + \mathbf{n}$$

به این منظور بردار فضای خالی^۱ در سمت چپ رابطه (۳-۸۰) ضرب می‌شود:

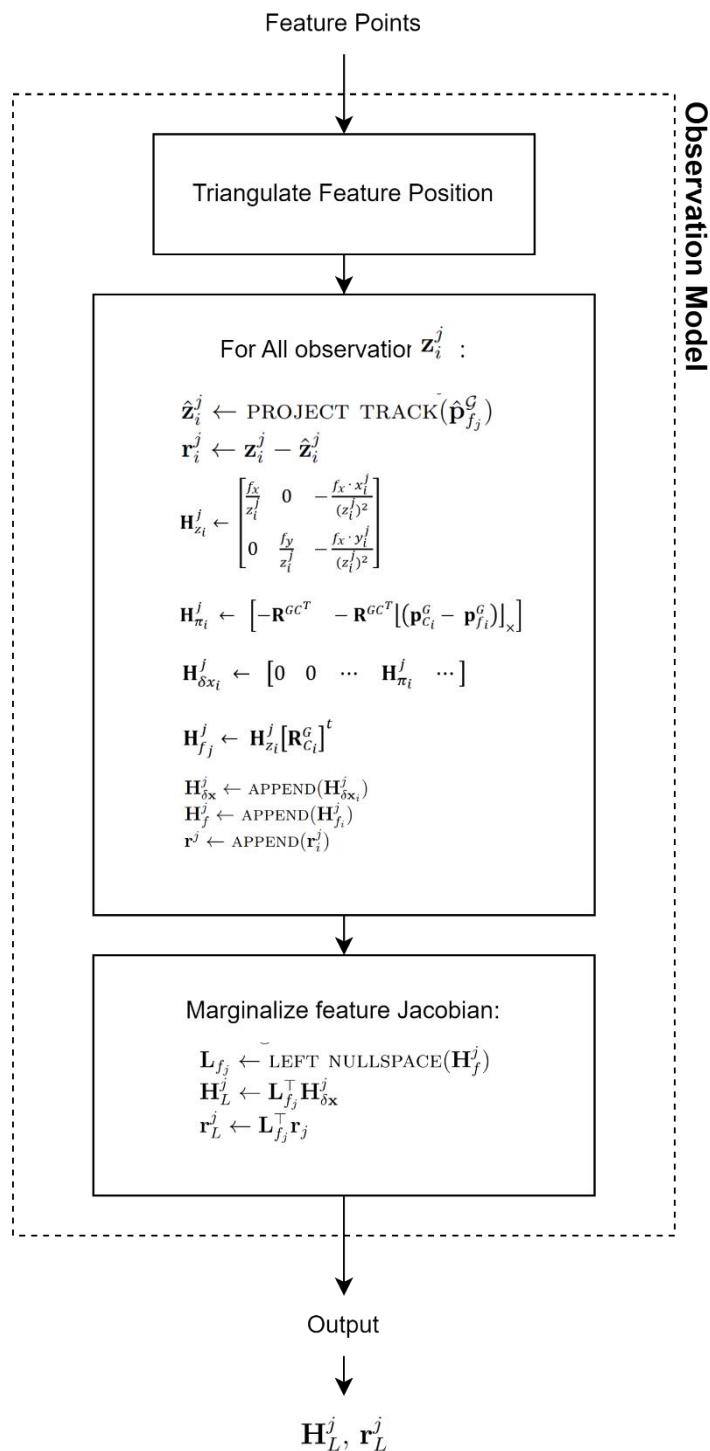
$$\begin{aligned} \mathbf{L}_{f_j}^T(\mathbf{r}^j) &= \mathbf{L}_{f_j}^T(\mathbf{H}_{\delta x}^j \delta x + \mathbf{H}_f^j \delta \mathbf{p}_{f_j} + \mathbf{n}) \\ \mathbf{L}_{f_j}^T &= \mathbf{L}_{f_j}^T \mathbf{H}_{\delta x}^j \delta x + \mathbf{L}_{f_j}^T \mathbf{n} \end{aligned}$$

که به صورت خلاصه می‌توان به صورت زیر نگارش کرد:

$$\mathbf{L}_L^i = \mathbf{H}_L^i \delta x + \mathbf{n}_L \quad (3-82)$$

ساختار کلی الگوریتم رؤیتگر را می‌توان در شکل ۱۸-۳ مشاهده کرد. در بخش بعدی بخش بروزرسان فیلتر بررسی خواهد شد.

Nullspace^۱



شکل ۱۸-۳ ساختار کلی بخش رؤیت‌گر

۶-۵-۳- توسعه بخش بروزرسان فیلتر^۱

آخرین قدم فیلتر MSCKF مرحله بروزرسانی می‌باشد که از قیود هندسی نقاط ویژه در جهت بروزرسانی فیلتر استفاده می‌شود تا خط رؤیت و اصلاح شود. در صورت وقوع دو شرط زیر مرحله بروزرسان باید اجرا شود:

- هنگام خارج شدن نقطه ویژه از پنجره دید دوربین
- هنگام کامل شدن ظرفیت پنجره کشویی (معمولاً یک اندازه بیشینه برای بردار پنجره کشویی درنظر گرفته می‌شود که از پرشدن حافظه سیستم جلوگیری شود).

در این مرحله نیاز است تا \mathbf{r}_L^j و \mathbf{H}_L^j بدست آمده در مرحله قبل در داخل دو ماتریس \mathbf{r}_L و \mathbf{H}_L جمع آوری شوند (فرض شود که تعداد نقاط ویژه موجود در تصویر K باشد):

$$\mathbf{H}_L = \begin{bmatrix} \mathbf{H}_L^0 \\ \mathbf{H}_L^1 \\ \vdots \\ \mathbf{H}_L^K \end{bmatrix} \quad \mathbf{r}_L = \begin{bmatrix} \mathbf{r}_L^0 \\ \mathbf{r}_L^1 \\ \vdots \\ \mathbf{r}_L^K \end{bmatrix} \quad (3-83)$$

بین دو ماتریس \mathbf{r}_L و \mathbf{H}_L رابطه زیر برقرار است:

$$\mathbf{r}_L = \mathbf{H}_L \delta x + \mathbf{n}_L \quad (3-84)$$

اکنون رابطه کلی فیلتر کالمن قابل اعمال می‌باشد که ماتریس کواریانس باقی‌مانده به صورت زیر در می‌آید:

$$\mathbf{Z} = \mathbf{H}_L \mathbf{P} \mathbf{H}_L^T + \mathbf{R}_L \quad (3-85)$$

در مرحله بعد، ضریب بهره فیلتر به صورت زیر محاسبه می‌شود:

$$\mathbf{K} = \mathbf{P} \mathbf{H}_L^T \mathbf{Z}^{-1} \quad (3-86)$$

در نهایت با داشتن ضریب بهره فیلتر کالمن، می‌توان خط رؤیت می‌شود:

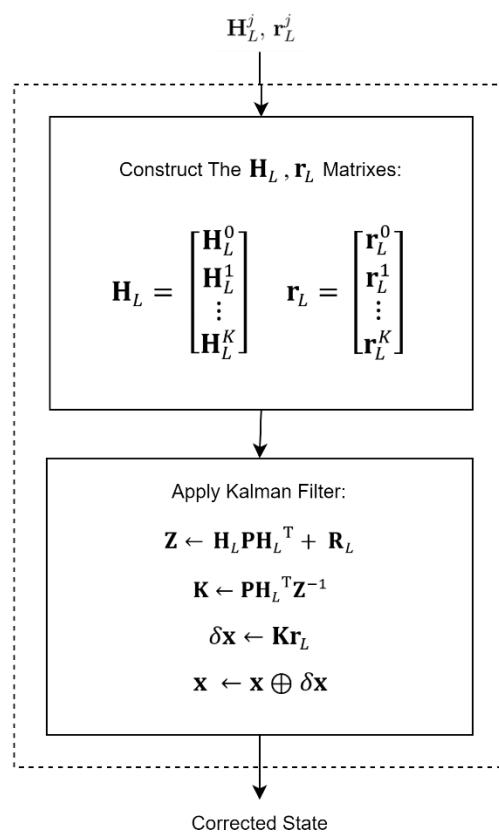
Filter Update ^۱

$$\delta\mathbf{x} \leftarrow \mathbf{K}\mathbf{r}_L \quad (3-87)$$

در نهایت با اضافه کردن این حالت خطأ (Error State) به حالت اسمی (Error State) می‌توان خطای انباشته شده را اصلاح کرد:

$$\mathbf{x} \leftarrow \mathbf{x} \oplus \delta\mathbf{x} \quad (3-87)$$

ساختار کلی بخش بروزرسان در شکل ۱۹-۳ قابل مشاهده است. در این فصل ساختار و جزئیات الگوریتم MSCKF ارایه شد. در فصل بعدی ابزارهای استفاده شده برای پیاده‌سازی الگوریتم و همچنین شبیه‌سازی آن معرفی و نحوه اجرای الگوریتم در محیط ROS ارایه خواهد شد.



شکل ۱۸-۳ ساختار کلی بخش بروزرسان

فصل چهارم

پیاده‌سازی الگوریتم طراحی مسیر فرود

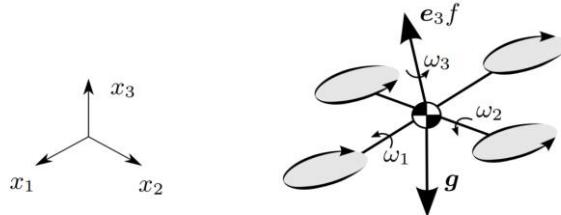
در فصل قبل مفاهیم لازم برای ایجاد یک سیستم موقعیت‌سنجی بصری-اینرسی مبتنی بر فیلتر MSCKF با جزئیات کامل ارایه شد. همانگونه در فصل مقدمه بیان شد، هدف از این پژوهش ایجاد یک سیستم موقعیت‌سنج بصری و استفاده از آن در کاربرد فرود آمدن پهپاد بر روی یک سکوی متحرک می‌باشد که در آن پهپاد دید مستقیم نسبت به سکوی متحرک ندارد (بخش ۱-۴). بدین منظور با استفاده از یک الگوریتم طراحی مسیر سه بعدی پیاده‌سازی شود تا با داشتن موقعیت لحظه‌ای پهپاد و سکوی متحرک، بتوان از آن برای هدایت پهپاد بهره برد. در این فصل ابتدا مدل‌سازی ریاضی پهپاد (با محوریت کوادکوپتر) بیان و سپس با در نظر گرفتن یکتابع هزینه، الگوریتم طراحی مسیر ارایه می‌شود. خروجی الگوریتم طراحی مسیر می‌تواند جهت کاربرد فرود آمدن پهپاد بر روی سکوی متحرک مورد استفاده قرار بگیرد.

۱-۴-۱- مدل‌سازی ریاضی پهپاد (کوادکوپتر)

قبل از پرداختن به الگوریتم طراحی مسیر نیاز است تا در ابتدا مدل ریاضی مربوط به پهپاد ارایه شود، در این پژوهش فرض می‌شود که پهپاد از نوع کوادکوپتر باشد. یک کوادکوپتر با فرض صلب بود می‌تواند ۶ درجه آزاد داشته باشد: سه درجه آزادی انتقالی و سه درجه آزادی زاویه‌ای. معمولاً برای بیان مدل ریاضی یک کوادکوپتر از دستگاه معادله دیفرانسیل زیر استفاده می‌شود:

$$\ddot{\mathbf{x}} = \mathbf{R}e_3 f + \mathbf{g} \quad , \quad \dot{\mathbf{R}} = \mathbf{R}[\boldsymbol{\omega}]_{\times} \quad (4-1)$$

که در آن \mathbf{x} بردار موقعیت کارتزین مرکز کوادکوپتر نسبت به دستگاه مرجع، \mathbf{R} ماتریس دوران کوادکوپتر نسبت به دستگاه مرجع، \mathbf{g} بردار شتاب گرانش، متغیر اسکالر f اندازه نیروی پیشران عمودی^۱ و $\boldsymbol{\omega}$ بردار سرعت زاویه‌ای می‌باشد (همانند شکل ۱-۴). همچنین بردار $(\mathbf{0}, \mathbf{0}, 1) = \mathbf{e}_3$ یک بردار یکه است در جهت محور x_3 می‌باشد.



شکل ۱-۴ نمایش چهار داده آخر پنجره کشویی [۶۱]

توجه شود که در معادله دیفرانسیل (۱-۴) متغیرهای f و $\boldsymbol{\omega}$ ورودی کنترلی کوادکوپتر هستند، این ورودی‌ها در کاربردهای گوناگون می‌توانند دارای محدودیت‌هایی باشند. برای مثال ممکن است حسگر سنجش اینرسی دارای حد اشیاع باشد و فقط تا محدوده خاصی از سرعت زاویه‌ای را بتواند اندازه‌گیری کند یا دوربین در تا حد اکثر مقدار معینی از نیروی پیشران عمودی بتواند بدستی فرآیند تصویر برداری را به صورت واضح انجام بدهد. بدین ترتیب به صورت زیر برای اندازه نیروی پیشران عمودی و بردار سرعت محدودیت در نظر گرفته می‌شود:

$$0 \leq f_{min} \leq f \leq f_{max} , \quad \|\boldsymbol{\omega}\| \leq \omega_{max} \quad (4-2)$$

اکنون فرض شود که بردار $(\mathbf{t}) \sigma$ حاوی متغیرهای حرکت انتقالی کوادکوپتر در فضای سه بعدی باشد که به ترتیب زیر تعریف می‌شود:

$$\sigma(t) = (\mathbf{x}(t), \dot{\mathbf{x}}(t), \ddot{\mathbf{x}}(t)) \quad (4-3)$$

زمانی گفته می‌شود که کوادکوپتر در زمان T به هدف $\hat{\sigma}$ رسیده است که تمام مولفه‌های بردار متغیرهای انتقالی کوادکوپتر با هدف یکی شود:

$$\sigma_i(t) = \hat{\sigma}_i \quad \forall i = \{1, 2, \dots, 9\} \quad (4-4)$$

هدف این فصل در واقع یافتن ورودی‌های کنترلی $(\mathbf{t}) f$ و $(\mathbf{t}) \boldsymbol{\omega}$ در بازه زمانی $[0, T]$ می‌باشد که کوادکوپتر را از یک شرایط اولیه مشخص (موقعیت، سرعت و شتاب اولیه) به هدف برساند $\hat{\sigma}$ و به عبارت دیگر تساوی (۴-۴) را بقرار کند. همچنین این ورودی‌های کنترلی باید به گونه‌ای باشد که روابط (۱-۴) و (۲-۴) نیز بقرار باشند.

۴-۲- الگوریتم طراح مسیر

برای یافتن بهینه‌ترین مسیر نیاز است تا یکتابع هزینه تعريف شود که با کمینه کردن آن، بهینه‌ترین مسیر محاسبه شود. معمولاً در این‌گونه از مسائل از انتگرال زیر استفاده می‌شود:

$$J_{\Sigma} = \frac{1}{T} \int_0^T \|j(t)\|^2 dt \quad (4-5)$$

که در انتگرال بالا $j(t)$ برابر تغییرات شتاب خطی نسبت به زمان می‌باشد به عبارت دیگر:

$$j = \ddot{x} = (\ddot{x}_1, \ddot{x}_2, \ddot{x}_3) \quad (4-6)$$

لازم به ذکر است که اندازه نیروی پیشران عمودی از اختلاف بین شتاب خطی و شتاب گرانش قابل محاسبه است:

$$f = \|\ddot{x} - g\| \quad (4-7)$$

با کمک روابط (۱-۴)، (۶-۴) و (۷-۴) می‌توان بردار تغییرات شتاب خطی را محاسبه کرد:

$$j = R[\omega]_x e_3 f + R e_3 \dot{f} \quad , \quad \dot{f} = e_3^T R^{-1} j \quad (4-8)$$

که بعد از ساده سازی معادله دیفرانسیل بالا به ترتیب زیر در می‌آید:

$$\begin{bmatrix} \omega_2 \\ -\omega_1 \\ 0 \end{bmatrix} = \frac{1}{f} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} R^{-1} j \quad (4-9)$$

توجه شود که در حالت حرکت خطی مقدار ω_3 برابر با صفر در نظر گرفته می‌شود، بدین ترتیب در محاسبات لحاظ نمی‌شود. رابطه (۹-۴) به همراه رابطه (۷-۴) یک دستگاه سه معادله-سه مجهول می-باشد که با داشتن تغییرات شتاب خطی j می‌توان ورودی‌های کنترلی f و ω را برای هر لحظه محاسبه و در نتیجه فرآیند طراحی مسیر را انجام داد. اکنون با داشتن رابطه (۹-۴) و (۵-۴) می‌توان فرآیند بهینه‌سازی را انجام داد. توجه شود که مقدار داخل انتگرال (۵-۴) سقف حاصل ضرب ورودی‌های کنترلی f و ω می‌باشد، به عبارت دیگر:

$$f^2 \|\omega\|^2 = \left\| f \begin{bmatrix} \omega_2 \\ -\omega_1 \\ 0 \end{bmatrix} \right\|^2 = \left\| \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} R^{-1} j \right\|^2 \leq \|j\|^2 \quad (4-10)$$

که با انتگرال گیری از طرفین نامساوی:

$$\frac{1}{T} \int_0^T f^2 \|\omega\|^2 dt \leq J_{\Sigma} \quad (4-11)$$

برای ساده‌تر شدن حل مسئله می‌توان $\sum J$ را در جهت سه محور به صورت جداگانه محاسبه کرد، به عبارت دیگر:

$$J_{\Sigma} = \sum_{k=1}^3 J_k \quad , \quad J_k = \frac{1}{T} \int_0^T j_k(t)^2 dt \quad (4-12)$$

در رابطه بالا J_k تصویر $\sum J$ بر روی محور k -ام می‌باشد که برای هر کدام از این محورها بردار حالت به صورت $(s_k, v_k, a_k) = (p_k, v_k, a_k)$ در می‌آید. برای اجرای فرآیند بهینه‌سازی از روش بهینه‌سازی پونتراچین^۱ استفاده می‌شود که در آن یکتابع همیلتونی تعریف شده و بهینه‌سازی براساس آن انجام می‌شود [۶۱]. در خروجی این بهینه‌سازی، تابع شدت تغییرات شتاب بهینه به صورت زیر در می‌آید:

$$j^*(t) = \frac{1}{2} \alpha t^2 + \beta t + \gamma \quad (4-13)$$

و با انتگرال‌گیری، بردار حالت مربوط به مسیر بهینه به صورت زیر در می‌آید:

$$s^*(t) = \begin{bmatrix} \frac{\alpha}{120} t^5 + \frac{\beta}{24} t^4 + \frac{\gamma}{6} t^3 + \frac{a_0}{2} t^2 + v_0 t + p_0 \\ \frac{\alpha}{24} t^4 + \frac{\beta}{6} t^3 + \frac{\gamma}{2} t^2 + a_0 t + v_0 \\ \frac{\alpha}{6} t^3 + \frac{\beta}{2} t^2 + \gamma t + a_0 \end{bmatrix} \quad (4-14)$$

که در آن شرایط ابتدایی به صورت $s(0) = (p_0, v_0, a_0)$ تعریف می‌شود و ضرایب α ، β و γ از رابطه زیر محاسبه می‌شود:

$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \frac{1}{T^5} \begin{bmatrix} 720 & -360T & 60T^2 \\ -360T & 168T^2 & -24T^3 \\ 60T^2 & -24T^3 & 3T^4 \end{bmatrix} \begin{bmatrix} p_f - p_0 - v_0 T - \frac{1}{2} a_0 T^2 \\ v_f - v_0 - a_0 T \\ a_f - a_0 \end{bmatrix} \quad (4-15)$$

که در آن $s(T) = (p_f, v_f, a_f)$ حالت نهایی کوادکوپتر است. پس در نتیجه با داشتن حالت (موقعیت، سرعت و شتاب خطی) در نقطه ابتدایی و نقطه نهایی می‌توان با کمک (۱۳-۴) و (۱۴-۴) مسیر بهینه را محاسبه کرد. همچنین با داشتن تغییرات شتاب خطی می‌توان ورودی‌های کنترلی f و ω از مجموعه روابط (۷-۴) و (۹-۴) را محاسبه کرد که با اعمال آن‌ها به کوادکوپتر مسیر بهینه طراحی شده، طی می‌شود.

Pontryagin^۱

فصل پنجم

پیاده‌سازی و بررسی نتایج

در فصل قبل تئوری و ساختار روش موقعیت‌سنگی بصری-اینرسی مبتنی بر فیلتر MSCKF و همچنین الگوریتم طراحی مسیر برای کوادکوپتر با جزئیات ارایه شد. اکنون با دانستن این پیش‌زمینه می‌توان به نحوه پیاده‌سازی یک الگوریتم موقعیت‌سنگ بصری-اینرسی پرداخت. در این فصل، ابتدا ابزارهای لازم برای پیاده‌سازی عملی همانند سیستم عامل ربات یا ROS [۶۲]، کتابخانه پردازش تصویر OpenCV [۶۳]، کتابخانه محاسبات جبری Eigen [۶۴] و ... بررسی خواهند شد. سپس نحوه پیاده‌سازی عملی الگوریتم MSCKF با ابزارهای ارایه شد، بیان خواهد شد. همچنین نحوه شبیه‌سازی کاربرد فرود آمدن پهپاد بر روی سکوی متحرک ارایه و در نهایت نتایج حاصل بررسی خواهد شد. لازم به ذکر است که کد و فایل‌های مربوط به این پژوهش به صورت متن باز در دسترس می‌باشد^۱.

۱-۵-۱- معرفی ابزارهای مورد نیاز

همان‌گونه که پیش‌تر بیان شد، در ابتدا ابزارهای مورد نیاز برای پیاده‌سازی الگوریتم موقعیت‌سنگی بصری-اینرسی و شبیه‌سازی آن در کاربرد فرود آمدن پهپاد بر روی سکوی متحرک ارایه خواهد شد. فهرست این ابزارهای به شرح زیر می‌باشد:

۱. سیستم عامل ربات یا ROS

GitHub Link: <https://github.com/ababayi/master-of-science-project> ^۱

۲. کتابخانه پردازش تصویر OpenCV

۳. کتابخانه حل معادلات جبر خطی Eigen

۴. نرم افزار شبیه‌ساز Gazebo

در ادامه هر کدام از ابزارهای ارایه شده در این فهرست، با جزئیات بیشتر بررسی خواهد شد.

۱-۵-۱- سیستم عامل ربات (ROS)

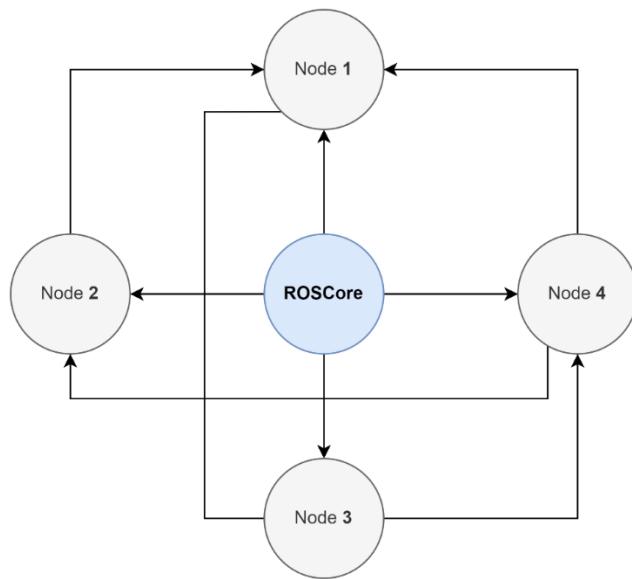
سیستم عامل ربات (ROS) یا Robot Operating System یک سکوی (یا چارچوب^۱) نرم افزاری قدرتمند است که معمولاً برای توسعه، کنترل، و مدیریت ربات‌ها و دستگاه‌های خودمختار به کار می‌رود. سیستم عامل ROS به عنوان یک چارچوب متن‌باز برای ربات‌ها شناخته می‌شود و که در ابتدا در سال ۲۰۰۹ توسط تیم Open Robotics توسعه یافت و اکنون توسط تیم Willow Garage توسعه و پشتیبانی می‌شود. این سیستم عامل به وسیله جامعه‌ای گسترشده از توسعه‌دهندگان به عنوان یک پلتفرم استاندارد به وجود آمده و امکان اشتراک‌گذاری کدها، توسعه پروژه‌های مختلف، و ایجاد کاربردهای رباتیکی را فراهم می‌کند. در فرآیند توسعه ربات‌ها، همانگونه که در شکل ۱-۵ نمایش داده شده است، نیاز است تا به سه بخش اصلی پرداخته شود: تنظیم حسگرها، پیاده‌سازی منطق ربات و پیکربندی عملگرها. هدف از ایجاد سیستم عامل ROS، مرکز کردن توسعه‌دهنده‌ها بر روی بخش منطق ربات می‌باشد، به عبارت دیگر توسعه‌دهنده‌ها دیگر نیاز به صرف زمان برای پیکربندی حسگرها و عملگرها ندارد و این دو بخش توسط سیستم عامل ROS انجام می‌شود (پکیج). در نتیجه توسعه‌دهنده می‌تواند در کمترین زمان ممکن، با کمک ابزارهای و کتابخانه‌های موجود در سیستم عامل ROS، کاربرد رباتیک مدنظر خود را به صورت بهینه اجرا کند.



شکل ۱-۵ فرآیند توسعه یک کاربرد رباتیک

به صورت کلی، از مزایای اصلی سیستم عامل ROS می‌توان به متن باز بودن، انعطاف‌پذیری، بروزرسانی منسجم، ابزارها و پکیج‌های آماده فراوان و پشتیبانی از انواع مختلف سخت‌افزارها برخوردار اشاره کرد که فرآیند توسعه و ایجاد ربات‌های پیشرفته را تسهیل می‌کند.

در سیستم عامل ROS یک برنامه (مطابق شکل ۲-۵) از تعدادی زیربخش تشکیل شده است که به هر کدام برنامه هرکدام از این زیربخش‌ها گره^۱ می‌گویند. گره‌ها واحدهای اجرایی در ROS هستند که هر کدام مستقل است که می‌تواند وظیفه‌های مختلفی را انجام دهد. گره‌ها می‌توانند با یکدیگر ارتباط برقرار کنند و اطلاعات را به اشتراک بگذارند. لازم به ذکر است که در سیستم عامل ROS یک گره مرکزی وجود دارد که به آن هسته^۲ سیستم عامل ROS می‌گویند که قبل از اجرای هر برنامه در سیستم عامل ROS باشستی این هسته فعال شود. این هسته وظیفه فراهم کردن محیط اجرایی و پکیج‌ها پیش‌نیاز، مدیریت کردن گره‌ها و ارتباط بین آن‌ها، مدیریت پارامترهای سیستم و ... بر عهده دارد و همه گره‌ها با آن در ارتباط هستند.

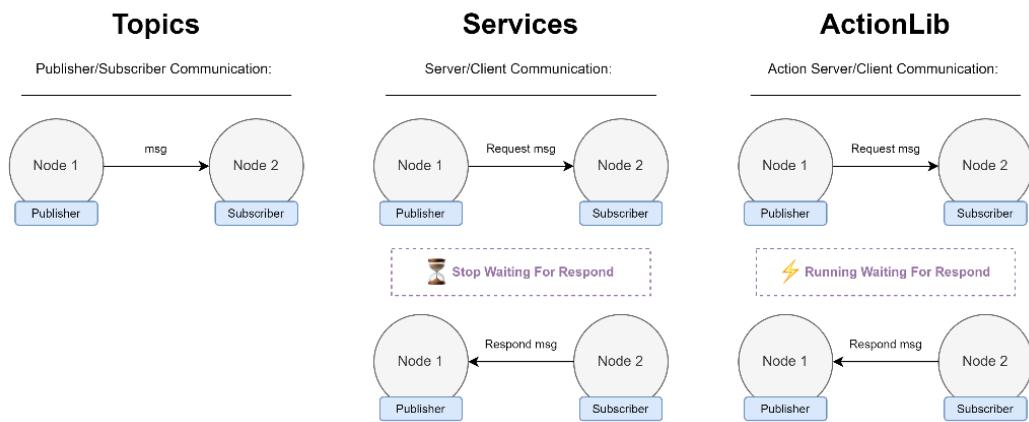


شکل ۲-۵ ساختار گره‌ای در سیستم عامل ROS

در سیستم عامل ROS، گره‌های موجود در برنامه ربات می‌توانند به سه روش مختلف با یکدیگر ارتباط برقرار کنند:

- روش ارتباطی Topics: در این روش به گره‌ها این امکان داده می‌شود تا داده‌ها را به صورت یک‌طرفه منتشر و دریافت کنند. به گره ارسال کننده داده‌ها، منبع منتشرکننده^۱ گویند و به گره‌هایی که آن اطلاعات را دریافت می‌کند، منبع دریافتکننده^۲ می‌گویند. این روش ارتباطی برای حالتی که قرار باشد داده به صورت یک‌طرفه و مدام (در فرکانس‌های بالا) ارسال شوند مناسب است.
- روش ارتباطی Services: این روش ارتباطی برخلاف روش Topics یک روش دو طرفه می‌باشد که در آن ابتدا گره کلاینت^۳ به گره سرور^۴ یک درخواست ارسال کرده و منتظر پاسخ سرور می‌ماند، در حین این انتظار برنامه کلاینت در حالت متوقف است. پس از دریافت پاسخ از گره سرور، کلاینت مجدد به فرآیند ادامه می‌دهد.
- روش ارتباطی ActionLib: این روش مشابه روش Service می‌باشد، با این تفاوت که پس از ارسال درخواست، برنامه تا دریافت پاسخ متوقف نمی‌ماند و فرآیندهای طراحی شده در گره کلاینت، همزمان با انتظار برای پاسخ گره سرور اجرا می‌شوند. برای درک بهتر نحوه کارکرد این سه روش ارتباطی، شکل شماره ۳-۵ را مشاهده نمایید.

Publisher^۱
Subscriber^۲
Client^۳
Server^۴



شکل ۳-۵ روش‌های ارتباطی بین گره‌ای در سیستم عامل ROS

لازم به ذکر است که در سیستم عامل ROS هر کدام از گره‌ها می‌توانند با یکی از زبان‌های برنامه‌نویسی (البته سیستم عامل ربات فقط از C++ و Python به صورت رسمی پشتیبانی می‌کند) ایجاد شود که این گره‌های ایجاد شده با زبان‌های مختلف، توانایی ارتباط با یکدیگر را از طریق سه روش ارتباطی ذکر شده دارند. همان‌گونه که پیش‌تر گفته شد، در داخل سیستم عامل ROS، پکیج و ابزارهای فراوانی ارایه شده است که برخی از معروف‌ترین این ابزارها عبارت‌اند از:

پکیج‌های roscpp و rospy

این دو ابزار برای استفاده از توابع و قابلیت‌های پیشفرض سیستم عامل ROS در داخل برنامه‌های مبتنی بر زبان C++ و Python ارایه شده است که فرآیندهایی همانند تعریف گره، ایجاد روش‌های ارتباطی گوناگون بین گره‌ای، کار با پیام‌ها و ... را می‌توان با کمک آنها انجام داد.

پکیج MoveIt

این پکیج یکی از ابزارهای بسیار معروف و قدرتمند در سیستم عامل ROS است که برای برنامه‌ریزی و مدیریت حرکتی در ربات‌ها مورد استفاده قرار می‌گیرد. این پکیج به توسعه‌دهندگان امکان ایجاد راهبردهای حرکتی برای ربات‌های صنعتی و تحقیقاتی را فراهم می‌کند.

پکیج Navigation

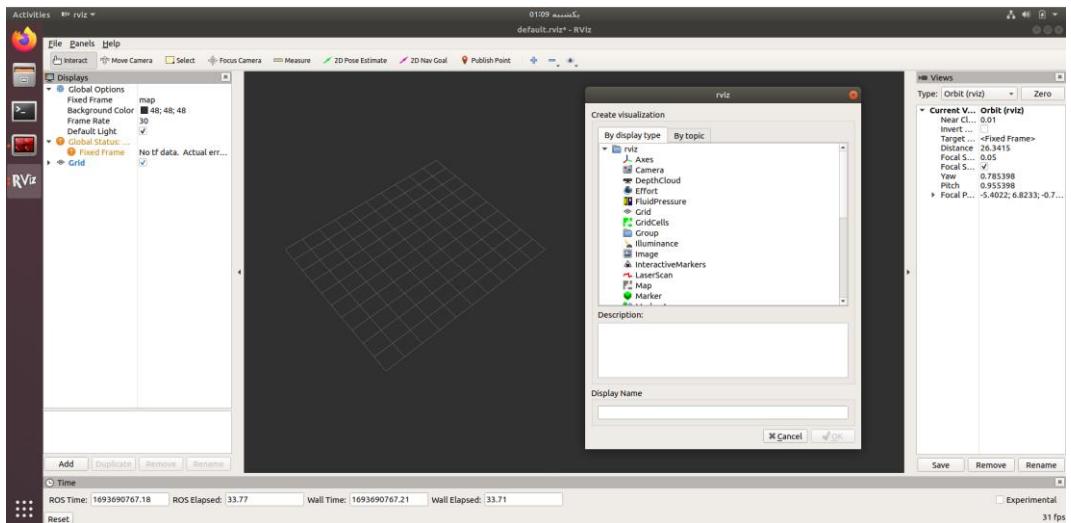
پکیج Navigation یکی دیگر از پکیج‌های مهم و پرکاربرد در سیستم عامل ROS است که برای مسیریابی و برنامه‌ریزی حرکت ربات‌ها (بالاخص موبایل ربات‌ها) در محیط‌های مختلف مورد استفاده قرار می‌گیرد. این پکیج به توسعه‌دهندگان امکان می‌دهد تا ربات‌ها را به گونه‌ای برنامه‌ریزی کنند که بتوانند به طور خودکار به اهداف خود برسند، از موانع پیش رو جلوگیری کنند و در محیط‌های ناشناخته مسیریابی کنند. از مزایا این پکیج می‌توان به سازگاری آن با روش SLAM اشاره کرد.

TF پکیج

یک پکیج بسیار مهم و کاربردی است که برای ایجاد و مدیریت تبدیل‌ها^۱ در محیط‌های رباتیکی استفاده می‌شود. تبدیل‌ها به توسعه‌دهندگان امکان مشخص کردن موقعیت و جهت‌گیری ربات‌ها و اجزاء مختلف در مختصات مرجع جهانی و نسبی را می‌دهند. با استفاده از پکیج TF امکان پیاده‌سازی دینامیکی ربات با کمک تبدیل‌ها فراهم می‌شود که به این ترتیب کنترل ربات میسر می‌شود.

Rviz پکیج

پکیج Rviz در سیستم عامل ROS یک ابزار بسیار کاربردی است که به توسعه‌دهندگان کمک می‌کند تا داده‌های مرتبط با ربات‌ها مانند داده‌های حسگرها و عملگرها را به صورت گرافیکی و سه‌بعدی نمایش دهند. این پکیج با امکاناتی نظیر تصویرسازی سه‌بعدی، انعطاف‌پذیری در پیکربندی، ارتباط زنده با داده‌های ربات و همچنین قابلیت شبیه‌سازی مقدماتی، یک ابزار کاربردی و چند منظوره در اکثر پروژه‌های رباتیک می‌باشد.



شکل ۴-۵ نمای کلی پکیج Rviz

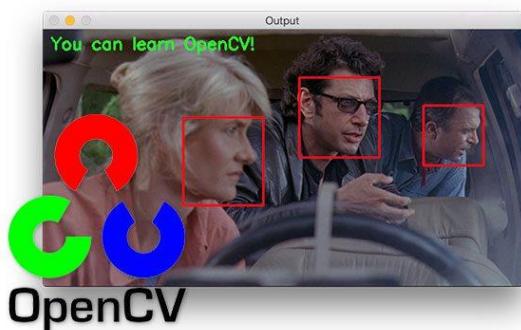
۱-۵-۲- کتابخانه پردازش تصویر OpenCV

کتابخانه پردازش تصویر OpenCV که به اختصار Open Source Computer Vision Library نامیده می‌شود، یک مجموعه قدرتمند از ابزار و توابع برنامه نویسی متن باز است که به تجزیه و تحلیل تصاویر و ویدیوها می‌پردازد. شروع توسعه کتابخانه پردازش تصویر OpenCV به سال ۱۹۹۹ برمی‌گردد و ابتدا توسط Intel توسعه یافت و منتشر شد. هدف اصلی ایجاد این کتابخانه، ایجاد یک چارچوب قوی و متن باز برای پردازش تصویر و بینایی ماشینی بود. این کتابخانه با پشتیبانی از متعددی زبان برنامه نویسی از جمله C++, Python و Java، به برنامه نویسان اجازه می‌دهد تا کاربردهای بینایی ماشین خود را در زمانی کوتاه انجام دهند. از دلایل مهم استفاده از این کتابخانه می‌توان به موارد زیر اشاره کرد:

- کارآیی بالا: کتابخانه OpenCV به عنوان یک چارچوب متن باز، بهینه‌سازی‌های زیادی برای پردازش تصویر دارد و عملکرد سریعی ارائه می‌دهد.
- پشتیبانی از فرمتهای متعدد: این کتابخانه قادر است تصاویر و ویدیوها را از فرمتهای مختلف تشخیص دهد و پردازش کند.
- قابلیت اجرای الگوریتم‌های پیشرفته: کتابخانه OpenCV با دسترسی به الگوریتم‌های پیشرفته و بروز، پیاده‌سازی اکثر کاربردهای بینایی ماشین در رباتیک، نظری تشخیص الگوها،

ردیابی اشیا، تطبیق ویژگی‌ها، تشخیص چهره، تشخیص وضعیت اجسام و ... را تسهیل می‌کند.

به طور کلی، OpenCV به عنوان یکی از ابزارهای حیاتی در زمینه پردازش تصویر و بینایی ماشینی شناخته می‌شود که به برنامه نویسان امکان می‌دهد تا در پروژه‌های خود از توانایی‌های قدرتمند و متعدد آن بهره‌برنده و به سرعت و به راحتی راهکارهای بصری پیشرفته را پیاده‌سازی کند.



شکل ۵-۵ اجرای کاربرد تشخیص چهره با کمک کتابخانه OpenCV [۶۵]

۱-۱-۵- کتابخانه حل معادلات جبر خطی Eigen

کتابخانه Eigen یک کتابخانه متن باز برای انجام عملیات جبر خطی در C++ می‌باشد که برای انجام محاسبات ماتریسی و برداری بسیار کارآمد و مفید طراحی شده است. این کتابخانه توسط گروهی از توسعه‌دهنگان نرم‌افزار در سال ۲۰۰۸ منتشر شد. دلیل اصلی برای ایجاد شدن Eigen این بود که نیاز به یک کتابخانه محاسبات جبری کارآمد و سریع در پروژه‌های نرم‌افزاری و علمی وجود داشت. از آن زمان تاکنون، Eigen به عنوان یکی از محبوب‌ترین و قدرتمندترین کتابخانه‌های جبر خطی در دنیای برنامه‌نویسی C++ شناخته می‌شود.

استفاده از کتابخانه Eigen بسیار مفید است زیرا به برنامه‌نویسان اجازه می‌دهد تا عملیات ماتریسی و برداری پیچیده را با سادگی و کارآیی بالا انجام دهند. این کتابخانه دارای تعداد زیادی الگوریتم ماتریسی مانند ضرب ماتریسی، تجزیه و تحلیل مقادیر ویژه، حل معادلات خطی، و غیره است. از این رو،

برنامه‌نویسان می‌توانند با استفاده از Eigen به سرعت پروژه‌های خود را پیاده‌سازی کنند و کارایی بالایی را در محاسبات جبری به دست آورند.

مزایای اصلی کتابخانه Eigen شامل عملکرد بالا، کد ساده و قابل فهم، پشتیبانی از متغیرهای مختلف (مانند اعداد صحیح و ممیزشناور)، و توانایی تعامل بسیار خوب با سایر کتابخانه‌ها و ابزارها در زمینه علم رباتیک می‌باشد. این کتابخانه در علم رباتیک بسیار کاربرد دارد، زیرا بسیاری از الگوریتم‌ها و مسائل در این حوزه به محاسبات جبری نیاز دارند، مانند محاسبه تبدیلات هندسی، کینماتیک و دینامیک ربات‌ها، پردازش تصویر و بینایی ماشینی و کنترل حرکتی. به عبارت دیگر، Eigen یک ابزار بسیار قوی برای تسهیل توسعه و پیاده‌سازی الگوریتم‌های رباتیک است که نیازمند محاسبات جبری هستند.

۱-۵-۳- شبیه‌ساز Gazebo

شبیه‌ساز Gazebo یکی از محبوب‌ترین ابزارهای شبیه‌سازی متن‌باز برای اجرای پروژه‌ها و تحقیقات در زمینه‌های رباتیک و کنترل خودمختار می‌باشد. این شبیه‌ساز در سال ۲۰۰۲ توسط نیت کوئنیگ^۱ توسعه یافت و به مرور سال‌ها توسط جامعه علمی و توسعه‌دهندگان در سرتاسر دنیا تکامل یافته است. شبیه‌ساز Gazebo به علت امکان شبیه‌سازی محیط‌های واقعی، امکان شبیه‌سازی انواع ربات‌ها و سنسورها و امکان انجام آزمایش‌های متعدد در یک محیط مجازی و ایمن، یکی از محبوب‌ترین شبیه‌سازهای حوزه رباتیک می‌باشد. این شبیه‌ساز می‌تواند به عنوان یک محیط آزمایشی قوی برای توسعه، تست، و آموزش الگوریتم‌های مختلف کنترل و هوش مصنوعی بکار رود. از مزایای آن می‌توان به کاهش هزینه‌ها و زمان مورد نیاز برای توسعه و تست سخت‌افزار و نرم‌افزار ربات‌ها، امکان ایجاد محیط‌های متنوع و قابل تنظیم برای آزمایشات، سازگاری کامل با سیستم عامل ROS و امکان اشتراک‌گذاری مدل‌ها و تجربیات با دیگر توسعه‌دهندگان اشاره کرد.

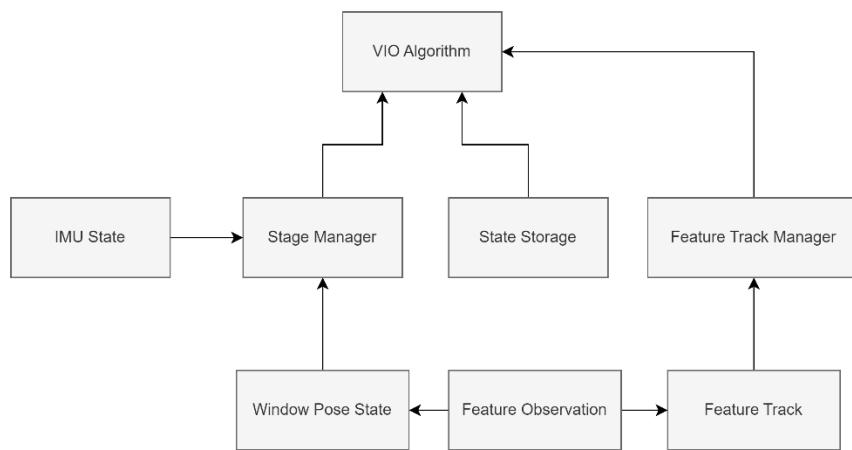
^۱ Nate Koenig



شکل ۶-۵ نمایی از شبیه‌سازی محیط صنعتی در Gazebo [۶۶]

۲-۵- پیاده سازی موقعیت‌سنجی بصری-اینرسی

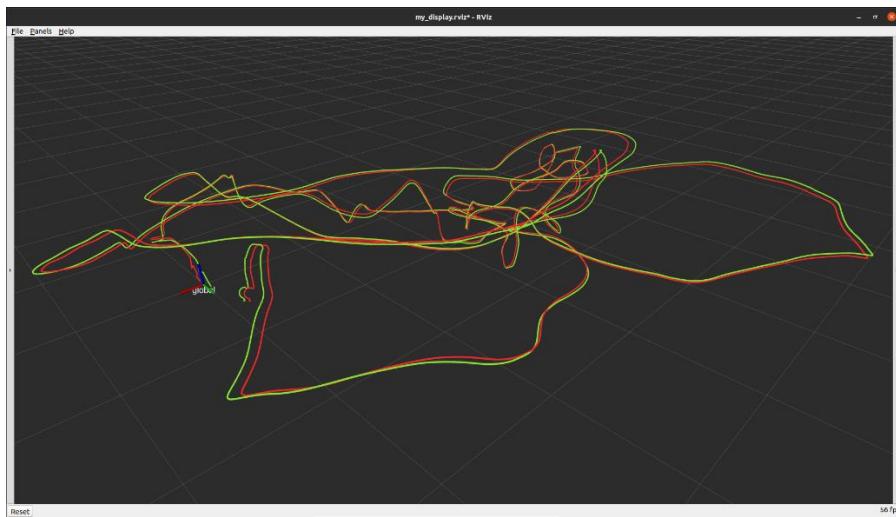
در فصل سوم در مورد تئوری‌ها و ساختار الگوریتم موقعیت‌سنجی بصری توضیح ارایه شد، در پیاده‌سازی عملی این الگوریتم نیز ساختار مشابهی نیاز است تا پیاده سازی شود. ساختار گره‌ای مربوط به الگوریتم موقعیت‌سنجی بصری-اینرسی پیاده شده در این پژوهش در شکل ۷-۴ قابل مشاهده است.



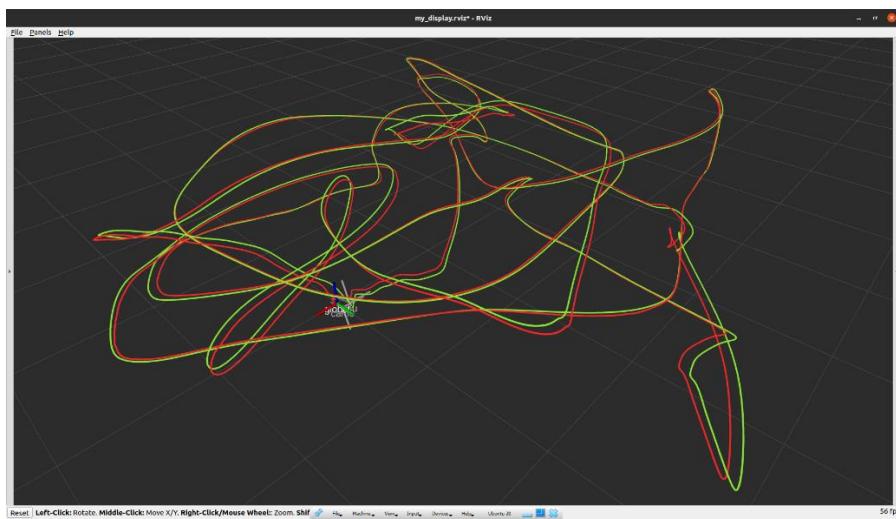
شکل ۷-۵ ساختار گره‌ای مربوط به الگوریتم موقعیت‌سنجی بصری-اینرسی

در شکل بالا گره مربوط به Feature Observation وظیفه شناسایی نقاط ویژه (گوشها) را با کمک کتابخانه OpenCV بر عهده دارد. گره‌های Feature Track Manager، Feature Track و Window Pose State نیز وظیفه ذخیره این نقاط ویژه در ماتریس کشویی و دنبال کردن آن‌ها را بر عهده دارند. گره IMU State وظیفه تخمین حالت حسگر سنجش اینرسی (IMU) و گره State Manager وظیفه تخمین

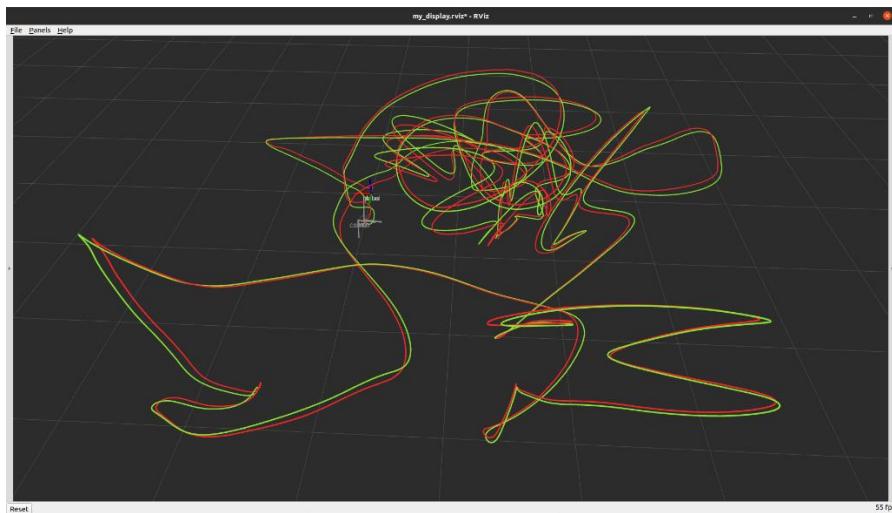
و محاسبه حالت خطرا برعهده دارد. همچنین در گره State Storage بردارهای حالت اسمی، خطوط و ماتریس کواریانس ذخیره می‌شوند. در نهایت در گره VIO Algorithm با کمک فیلتر کالمن MSCKF موقعیت‌سنگی بصری-اینرسی و بروزرسانی‌های لازم برای فریم بعدی انجام می‌شود. خروجی الگوریتم بالا بر روی دیتابست استاندارد EuRoC MAV [۶۷] در شکل‌های (۷-۵)، (۸-۵) و (۹-۵) نمایش داده شده است.



شکل ۷-۵ تست الگوریتم موقعیت‌سنگی بر روی دیتابست V1_01_easy



شکل ۸-۵ تست الگوریتم موقعیت‌سنگی بصری-اینرسی بر روی دیتابست V1_02_medium



شکل ۹-۵ تست الگوریتم موقعیت‌سنجی بصری-اینرسی بر روی دیتابست V1_03_difficult

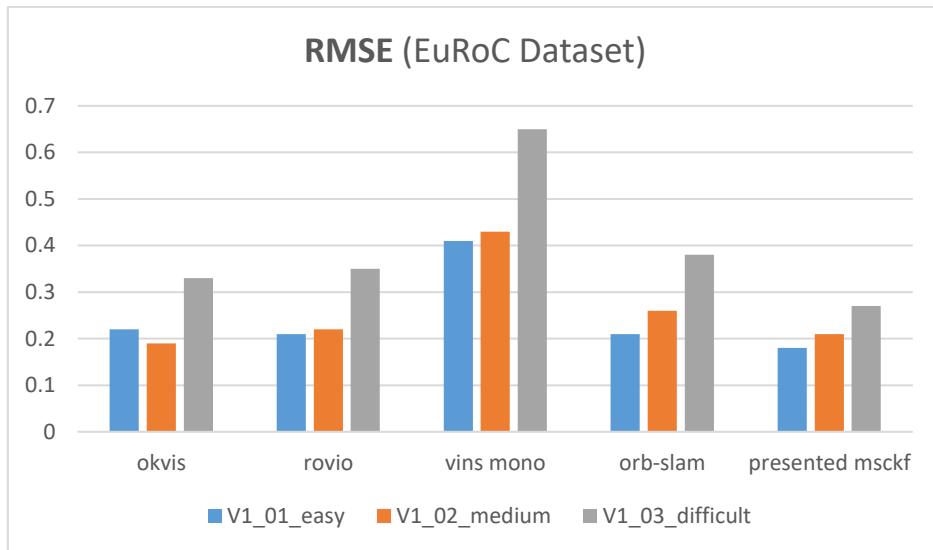
توجه شود که در شکل‌های (۷-۵)، (۸-۵) و (۹-۵) مسیر قرمز رنگ، مسیر تخمین زده شده توسط الگوریتم موقعیت‌سنج MSCKF و مسیر سبز رنگ موقعیت صحیح می‌باشد. تمامی تست‌های بر روی یک ماشین مجازی^۱ با منابع ۱ هسته پردازشی (۱/۲ گیگاهرتز) و ۲ گیگابایت حافظه RAM انجام شده

است که جزء حداقل ترین منابع پردازشی در کامپیوترهای کوچک می‌باشد. برای سنجش دقیق الگوریتم‌های موقعیت‌سنج از یک معیار با عنوان ریشه میانگین مربعات خطأ^۲ استفاده می‌شود که به صورت خلاصه RMSE نامیده می‌شود و به کمک رابطه زیر محاسبه می‌شود:

$$RMSE = \sqrt{\frac{\sum_1^N (x_{True} - x_{Observed})^2}{N}} \quad (5-1)$$

که در رابطه بالا N تعداد مشاهدات (اینجا تعداد نقاط مسیر طی شده) می‌باشد. این معیار برای الگوریتم موقعیت‌سنج ارایه شده محاسبه شده و در شکل ۱۰-۵ در مقایسه با سایر روش‌ها به نمایش گذاشته شده است.

Virtual Machine
Root Mean Square Error

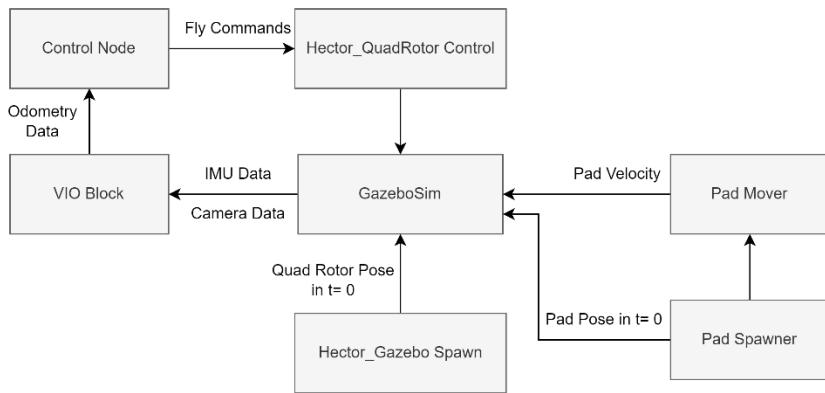


شکل ۵-۱۰ مقایسه الگوریتم های مختلف موقعیت‌سنجی براساس معیار RMSE

همانگونه که در شکل ۵-۱۰ قابل مشاهده است، الگوریتم موقعیت‌سنجی MSCKF عالرقم برخوردار بودن از منابع پردازشی محدود، دارای دقت مناسب در مقایسه با سایر روش‌ها می‌باشد که استفاده از آنرا در کاربردهای مختلف از جمله فرود آمدن پهپاد بر روی سکوی متحرک، مناسب می‌کند.

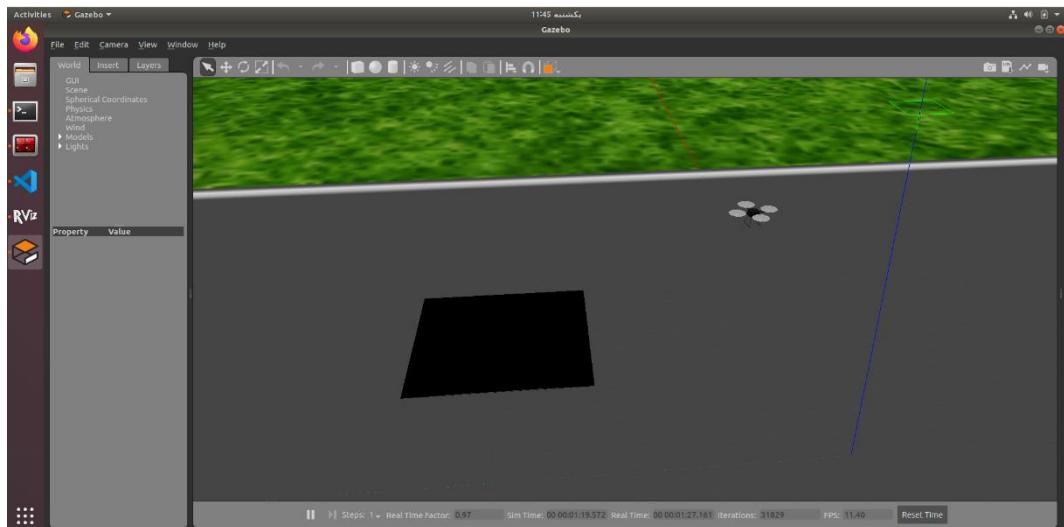
۳-۵-۳- پیاده سازی کاربرد فرود آمدن پهپاد بر روی سکوی متحرک

در بخش مقدمه ذکر شد که یکی از اهداف این پژوهش شبیه‌سازی کاربرد فرود آمدن پهپاد بر روی سکوی متحرک می‌باشد که در آن پهپاد، موقعیت سکو را نسبت به مرجع مختصات مرجع می‌داند و هیچ دید مستقیمی نسبت به آن ندارد. همچنین از الگوریتم موقعیت‌سنجی بصری اینرسی برای تخمین موقعیت خود استفاده می‌کند. برای شبیه‌سازی این کاربرد از ترکیب سیستم عامل ROS و شبیه‌ساز Gazebo استفاده می‌شود. ساختار گره‌ای برنامه این شبیه‌سازی در شکل ۸-۵ نمایش داده شده است.



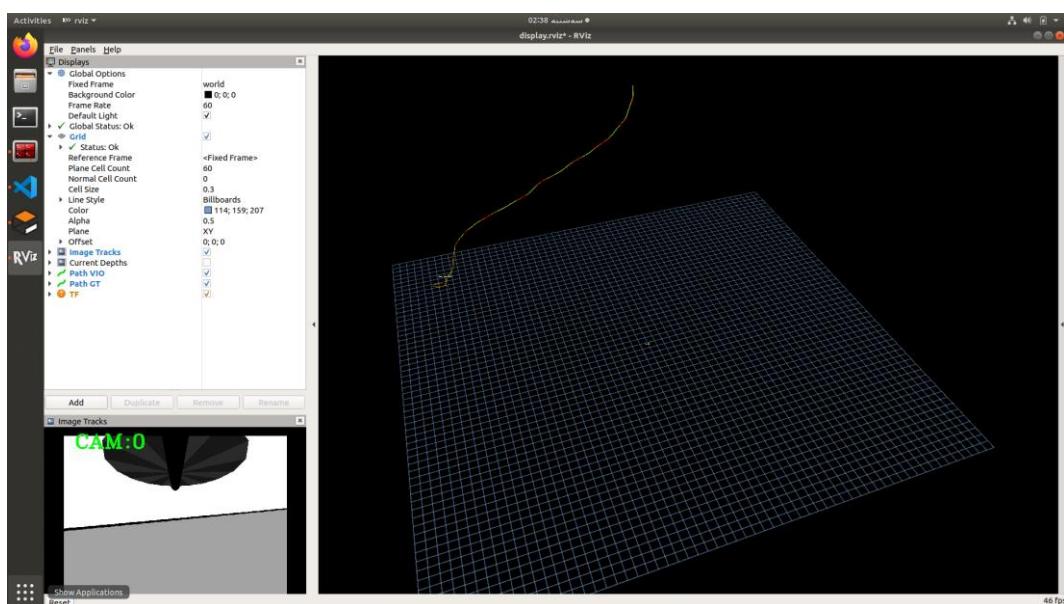
شکل ۱۱-۵ ساختار کلی گره‌های شبیه‌سازی

در شکل بالا گره GazeboSim وظیفه اتصال بین سیستم عامل ROS و شبیه‌ساز Gazebo را بر عهده دارد و براساس داده‌های مشخص شده می‌تواند اجزا جدید به محیط شبیه‌سازی اضافه و یا کم کرد. همچنین این گره وظیفه دارد تا داده‌های حسگرها را در قالب Topic درون هسته ROS منتشر کند و همچنین در صورت نیاز عملگرهای موجود در ربات شبیه‌سازی شده را از طریق روش‌های ارتباطی استاندارد ROS به کار بیندازد. همچنین گره Pad Mover و Pad Spawner وظیفه اضافه کردن سکوی متحرک به محیط شبیه‌ساز و حرکت دادن آنرا بر عهده دارند. بلوك VIO Block حاوی گره‌های مربوط به موقعیت‌سنجی بصری-اینرسی MSCKF که در شکل ۷-۵ ارایه شده است، می‌باشد. گره Hector_Gazebo Spawn یک گره از پکیج استاندارد Hector (یک پکیج برای شبیه‌سازی کوادکوپتر) است و وظیفه آن اضافه کردن یک کوادکوپتر به محیط شبیه‌ساز است. همچنین گره Hector_QuadRotor Control وظیفه حرکت دادن پهپاد با استفاده از فرمان‌های ورودی را بر عهده داد. در شکل ۱۰-۵ گره Controller نیز وظیفه دریافت و پردازش موقعیت‌سنجی بصری و همچنین موقعیت سکوی متحرک و در نتیجه ارایه فرمان مناسب برای پهپاد در جهت فرود آمدن بر روی سکو را بر عهده دارد. نمایی از خروجی این شبیه‌سازی در شکل ۱۱-۵ قابل مشاهده است.



شکل ۱۲-۵ نمایی از شبیه‌سازی کاربرد فرود آمدن پهپاد بر روی سکوی متحرک

لازم به ذکر است که برای پیاده‌سازی بخش کنترل کننده فرود، از مرجع [۶۱] استفاده شده است که امکان طراحی مسیر برای پهپادها با داشتن موقعیت اولیه و نهایی را با لحاظ محدودیت (در صورت نیاز) فراهم می‌کند، که مبانی لازم آن در فصل چهارم ارایه شد. نمایی از مسیر طی شده فرود شبیه‌سازی شده در شکل ۱۰-۴ قابل مشاهده است (خط سبز رنگ مقدار واقعی موقعیت و مقدار قرمز رنگ مقدار تخمین زده شده است).



شکل ۱۰-۴ مسیر طی شده توسط پهپاد برای فرود آمدن بر روی سکوی متحرک

این شبیه‌سازی چندین بار تکرار شده و مقدار پارامتر RMSE محاسبه گردیده است، به صورت میانگین این پارامتر در حدود 21 ± 0 اندازه‌گیری شده که خطای قابل قبولی است و این خطای محدود در فرآیند کنترل فرود آمدن پهپاد بر روی سکوی متحرک قابل چشم‌پوشی است.

فصل ششم

نتیجه‌گیری و پیشنهادات

در این پژوهش در ابتدا اهمیت و کلیدی بودن موقعیت‌سنجی در سیستم‌های خودمختار تبیین و روش‌های موقعیت‌سنجی مختلف همانند موقعیت‌سنجی با GPS، لیزر اسکنر، لیدار، دوربین تکی یا استریوو ارایه و بررسی شد. همچنین برتری روش موقعیت‌سنجی بصری-اینرسی نسبت به سایر روش‌ها به دلیل چابک و ارزان بودن، انعطاف‌پذیری بالا، دقیق مناسب و ... برای کاربردهای پهپادی مشخص شد.

در مرحله بعد، پس از ارایه مقدماتی در مورد حسگر سنجش اینرسی (IMU) و نحوه مدل‌سازی آن، فیلتر کالمن مبتنی بر خطای حالت ارایه شد که در آن نامعینی‌ها برخلاف فیلتر کالمن توسعه یافته معمولی، بجای بردار حالت اسمی در بردار خطای حالت انعکاس داده می‌شوند.

پس از آن مبانی بینایی ماشین لازم برای تشخیص نقاط ویژه و دنبال کردن آنها درون تصویر ارایه شد. در این بخش از گوشه‌های موجود در تصویر به عنوان نقاط ویژه استفاده شد که این گوشه‌ها علاوه بر سهولت در شناسایی، قابلیت یافتن مجدد در فریم‌های بعدی را نیز فراهم می‌کند. همچنین در پایان بخش مقدمات بینایی ماشین، درباره روش مثلث‌گیری برای تخمین موقعیت گوشه‌ها در فضای سه بعدی توضیحات لازم ارایه شد.

در بخش بعدی تئوری و ساختار الگوریتم موقعیت‌سنجی بصری-اینرسی MSCKF ارایه شد. این الگوریتم با توجه به استفاده از یک پنجره کشوبی برای ذخیره داده‌های قدیمی و یک الگوریتم بروزرسان برای اصلاح خطاهای آن، دارای دقت مناسبی است. همچنین با توجه به محور قرار دادن حسگر سنجش اینرسی (IMU) به عنوان حسگر اصلی و دوربین به عنوان حسگر فرعی، دارای محاسبات بسیار کمتر در مقایسه با سایر روش‌های موقعیت‌سنجی بصری-اینرسی می‌باشد.

در نهایت این الگوریتم با کمک کتابخانه‌های مشهور OpenCV و Eigen تحت سیستم عامل ROS پیاده‌سازی شد. همانگونه که در بخش پیاده‌سازی ارایه شد، این الگوریتم می‌تواند با تعداد نقاط ویژه محدود بر روی سخت‌افزارهای ارزان قیمت (نظیر کامپیوتراهای کوچک) استفاده شود. البته باید در نظر گرفت که انتخاب تعداد نقاط ویژه یک سقف دارد که این سقف توسط منابع پردازشی کامپیوتر مشخص می‌شود. در پایان بخش پیاده‌سازی با کمک شبیه‌ساز Gazebo کاربرد فرود آمد پهپاد بر روی سکوی متحرک (در شرایطی که پهپاد دید مستقیم به سکوی متحرک ندارد) ارایه شد که در آن برای موقعیت-سنجی پهپاد از الگوریتم موقعیت‌سنجی بصری-اینرسی MSCKF بهره برداری شد.

لازم به ذکر است که به علت محدود بودن منابع مالی و زمانی، کاربرد فرود آمدن پهپاد بر روی سکوی متحرک، در بستر شبیه‌سازی انجام شد. پیاده‌سازی این کاربرد در دنیای واقعی با در نظر گرفتن شرایط مذکور می‌تواند در ادامه این تحقیق مورد توجه باشد.

منابع و مأخذ

- [1] "Autonomous robot," [Online]. Available: https://en.wikipedia.org/wiki/Autonomous_robot.
- [2] "Simultaneous localization and mapping," [Online]. Available: https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping.
- [3] "Odometry," [Online]. Available: <https://en.wikipedia.org/wiki/Odometry>.
- [4] A. B. by Jonathan Seybold, "Miniaturized Optical Encoder with Micro Structured Encoder Disc," *MPDI*, 2019.
- [5] "hokuyo sensors," hokuyo, [Online]. Available: <https://www.hokuyo-aut.jp/search/single.php?serial=169>.
- [6] "yahboom sesnors," yahboom, [Online]. Available: https://category.yahboom.net/products/rplidar_slamtec.
- [7] Wikipedia, "Visual odometry," [Online]. Available: https://en.wikipedia.org/wiki/Visual_odometry.
- [8] S. A. R. Mohamed Aladem, "Lightweight Visual Odometry for Autonomous Mobile Robots," *MPDI*, 2019.
- [9] "Stereo Camera," waveshare, [Online]. Available: https://www.waveshare.com/wiki/IMX219-83_Stereo_Camera.
- [10] Y. M. Wang, "A camera calibration technique based on OpenCV," *IEEE*, 2010.
- [11] "Lucas–Kanade method," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Lucas%E2%80%93Kanade_method.
- [12] S. R. Gonzalez, "Control of off-road mobile robots using visual odometry," *Control of off-road mobile robots using visual odometry*, 2013.
- [13] D. McManus, "Towards lighting-invariant visual navigation: An appearance-based," *Robotics and Autonomous Systems*, 2013.
- [14] H. E. Benseddik, "SIFT and SURF Performance Evaluation for Mobile Robot-Monocular Visual," *International Journal of Image and Graphics*, 2014.
- [15] S. D. Pizzoli, "REMODE: Probabilistic, monocular dense reconstruction in real time," *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [16] C. D. Enge, "LSD-SLAM: Large-Scale Direct Monocular SLAM," *Lecture Notes in*, 2014.

- [17] S. D. Forster, "SVO: Fast semi-direct monocular visual odometry," *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [18] T. J. D. Mur-Artal, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, 2015.
- [19] C. B. Heng, "Semi-direct visual odometry for a fisheye-stereo camera," *International Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [20] P. Liu, "Direct Visual Odometry for a Fisheye-Stereo Camer," *International Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [21] Y. Feng, "A fusion algorithm of visual odometry based on feature-based method and direct," *Chinese Automation Congress (CAC)*, 2017.
- [22] B. S. Krombach, "Combining Feature-Based and Direct Methods for Semi-dense Real-Time Stereo Visual Odometry," *Advances in Intelligent Systems and Computing*, 2017.
- [23] K. A. Lee, "Event-based real-time optical flow estimation," *International Conference on Control, Automation and Systems (ICCAS)*, 2017.
- [24] S. D. Forster, "Semidirect Visual Odometry for Monocular and Multicamera Systems," *IEEE Transactions on Robotics*, 2017.
- [25] Y. Zhou, "Canny-VO: Visual Odometry with RGB-D Cameras," *IEEE TRANSACTIONS ON ROBOTICS*, 2020.
- [26] C. D. Engel, "Direct Sparse Odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.
- [27] G. Schlegel, "Transactions on Pattern Analysis and Machine Intelligence," *International Conference on Robotics and Automation (ICRA)*, 2018.
- [28] Y. Liu, "Accurate and Robust Monocular SLAM with Omnidirectional Cameras," *Sensors*, 2019.
- [29] C. R. Muñoz-Salinas, "UcoSLAM: Simultaneous localization and mapping by fusion of keypoints and squared planar markers," *Pattern Recognition*, 2020.
- [30] W. Guan, "Visual Inertial Odometry," *IEEE*, 2021.
- [31] F. P. Leutenegger, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, 2014.
- [32] S. D. Forster, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, 2017.
- [33] S. D. Kaiser, "Simultaneous State Initialization and Gyroscope Bias Calibration in

Visual Inertial Aided Navigation," *IEEE Robotics and Automation Letters*, 2017.

- [34] S. R. Bloesch, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, 2017.
- [35] K. V. Sun, "Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight," *IEEE Robotics and Automation Letters*, 2018.
- [36] T. Qin, "A General Optimization-based Framework for Global," *IEEE*, 2019.
- [37] C. Campos, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM," *Computer Science*, 2021.
- [38] A. I. & R. Mourikis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," *IEEE International Conference on Robotics and Automation*, 2015.
- [39] A. Kelly, *Mobile Robotics*, Cambridge, 2014.
- [40] J. Sol`a, "Quaternion kinematics for the error-state Kalman filter," *arXiv*, 2017.
- [41] "Quaternion," wikipedia, [Online]. Available:
<https://en.wikipedia.org/wiki/Quaternion>.
- [42] "Euler method," wikipedia, [Online]. Available:
https://en.wikipedia.org/wiki/Euler_method.
- [43] "Kalman filter," wikipedia, [Online]. Available:
https://en.wikipedia.org/wiki/Kalman_filter.
- [44] "Computer vision," wikipedia, [Online]. Available:
https://en.wikipedia.org/wiki/Computer_vision.
- [45] M. S. C. G. Harris, "A Combined Corner and Edge Detector," *Semantic Scholar*, 1998.
- [46] "opencv harris corner detector," datahacker, [Online]. Available:
<https://datahacker.rs/opencv-harris-corner-detector-part1/>.
- [47] "Harris corner detector," wikipedia, [Online]. Available:
https://en.wikipedia.org/wiki/Harris_corner_detector.
- [48] "Sobel operator," wikipedia, [Online]. Available:
https://en.wikipedia.org/wiki/Sobel_operator.
- [49] "Optical flow," wikipedia, [Online]. Available:
https://en.wikipedia.org/wiki/Optical_flow.
- [50] "tutorial optical flow," opencv, [Online]. Available:
https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html.

- [51] T. K. Bruce D. Lucas, "An Iterative Image Registration Technique with an Application to Stereo Vision (IJCAI)," *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, 1981.
- [52] S. Roy, "Reconstruction of a class of fluid flows by variational methods and inversion of integral transforms in tomography.," *University of Texas at Arlington*, 2015.
- [53] "Weighted least squares," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Weighted_least_squares.
- [54] "Cross correlation," wikipedia, [Online]. Available: <https://en.wikipedia.org/wiki/Cross-correlation>.
- [55] "Pinhole camera," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Pinhole_camera.
- [56] H. Opower, "Multiple view geometry in computer vision," *Optics and Lasers in Engineering*, 2002.
- [57] "camera calibration using opencv," learnopencv, [Online]. Available: <https://learnopencv.com/camera-calibration-using-opencv/>.
- [58] R. Heale, "Understanding triangulation in research," *BMJ*, 2018.
- [59] B. H. Otmar Scherzer, "Gauss–Newton method for solving linear inverse problems with neural network coders," *Springer*, 2023.
- [60] "Linear least squares," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Linear_least_squares.
- [61] M. W. H. M. & D. R. Mueller, " A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation," *IEEE Transactions on Robotics*, 2015.
- [62] "ros," ros, [Online]. Available: ros.org.
- [63] "opencv," opencv, [Online]. Available: <https://opencv.org/>.
- [64] "eigen," eigen, [Online]. Available: https://eigen.tuxfamily.org/index.php?title=Main_Page.
- [65] "opencv tutorial a guide to learn opencv," pyimagesearch, [Online]. Available: <https://pyimagesearch.com/2018/07/19/opencv-tutorial-a-guide-to-learn-opencv/>.
- [66] "Gazebo simulator," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Gazebo_simulator.
- [67] "Dataset," ethz, [Online]. Available: <https://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets>.

Abstract

Today, with the expansion of knowledge in various fields of science and technology, the field of robotics is developing day by day with the aim of increasing productivity and optimization. One of the most important goals of this field is to reduce or completely eliminate human intervention in carrying out processes; Because human intervention in many processes causes errors due to low accuracy, fatigue, etc. For this purpose, a new field called autonomous robots has been created with the aim of reducing or eliminating human intervention. Usually, the structure of the planning system of these robots consists of three main parts: map creation, localization and route design, and the robot needs an accurate odometry of its location for all three stages. In this way, odometry has a special importance in autonomous robots. This odometry can be done in different ways such as using GPS, laser scanner, lidar, single or stereo camera, etc. These methods all have many advantages, but some are not suitable for some robotic applications such as drones due to their high cost (lidar and laser scanner) and some due to low accuracy or calibration problems (GPS and camera). For this purpose, the visual-inertial odometry method is used, which is created by combining an inertial measurement sensor (IMU) and one or more cameras. In addition to being cheap and agile, this method has high flexibility for use in most robotic applications such as drones. In this research, the aim is to create a visual inertial odometry system based on the multi-state constraint Kalman filter (MSCKF) that can be used in the application of UAV landing on a moving platform. In this research, first the basic concepts required for the development of this algorithm such as sensor modeling, machine vision preliminaries, error-state Kalman filter are presented and then these concepts will be used to develop the related odometry method. At the end, this algorithm is implemented with the help of the famous OpenCV and Eigen libraries in the platform of the robot operating system (ROS) and it will be used in the application of the landing of the drone on the moving platform in the form of simulation. It should be mentioned that Gazebo simulator software is used for the application of UAV landing on the moving platform.

Keywords: Autonomous Systems – Odometry – Visual Inertial Odometry – Tracking Moving Target – Robot Operating System (ROS)

در هنگام بارگذاری در سامانه سپندای این صفحه حذف شود



Vice Chancellor for Research and Technology
Department of Engineering
Faculty of Mechanical Engineering
University of Isfahan

A thesis entitled

Design and development of a visual inertial odometry system with moving target tracking approach for autonomous robots

was submitted by

Ahmad Babaei Bondarti

in partial fulfillment of the requirement for the award of the degree of Master of Science.
The degree, evaluated as "excellent/ very good/ good/ intermediate", was awarded on 01-
01-2023

by the following examination committee.

	Full name	Academic rank	Institute	Signature
1. Supervisor	Dr. Hossein Karimpour	Associate Professor	University of Isfahan	
2. Co-Supervisor	Dr. ...	Professor (Full)	University of Isfahan	
3. Advisor	Dr. ...	Assistant Professor	University of Isfahan	
4. Internal assessor	Dr. ...	Associate Professor	University of Isfahan	
5. External assessor	Dr. ...	Associate Professor	University of ...	

Head of the department

Name and signature:



University of Isfahan
Faculty of Engineering
Department of Mechanical Engineering

M.Sc. Thesis

**Design and development of a visual inertial odometry system
with moving target tracking approach for autonomous robots**

Supervisor/Supervisors:

Dr. Hossein Karimpour

Advisor/Advisors:

By:

Ahmad Babaei Bondarti

September 2023