



دانشکده مهندسی کامپیوتر

پیاده سازی ابزار تبدیل خودکار تصویر نمونه به مرحله آماده برای موتور بازی سازی یونیتی .

پایاننامه برای دریافت درجه کارشناسی
در رشته مهندسی کامپیوتر

هانا کریمان

استاد راهنما:

دکتر بهروز مینایی

1402 بهمن ماه

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ

تأییدیه‌ی هیئت‌داوران جلسه‌ی دفاع از پایان‌نامه/رساله

نام دانشکده: دانشکده مهندسی کامپیوتر

نام دانشجو: هانا کریمان

عنوان پایان‌نامه: پیاده سازی ابزار تبدیل خودکار تصویر نمونه به مرحله آماده برای موتور بازی سازی یونیتی

تاریخ دفاع:

رشته: مهندسی کامپیوتر

ردیف	سمت	نام و نام خانوادگی	مرتبه دانشگاهی	دانشگاه یا مؤسسه	امضا
1	استاد راهنما	دکتر بهروز مینایی	استادیار	دانشگاه علم و صنعت ایران	
2	استاد مدعو داخلی		دانشیار	دانشگاه علم و صنعت ایران	
3	استاد مدعو خارجی		استادیار		

تأییدیه‌ی صحت و اصالت نتایج

با اسمه تعالی

این جانب هانا کریمان به شماره دانشجویی 97522283، دانشجوی رشته مهندسی کامپیوتر مقطع تحصیلی کارشناسی تأیید می‌نمایم که کلیه‌ی نتایج این پایان‌نامه/رساله حاصل کار این جانب و بدون هرگونه دخل و تصرف است و موارد نسخه‌برداری شده از آثار دیگران را با ذکر کامل مشخصات منبع ذکر کرده‌ام. در صورت اثبات خلاف مندرجات فوق، به تشخیص دانشگاه مطابق با ضوابط و مقررات حاکم (قانون حمایت از حقوق مؤلفان و مصنفان و قانون ترجمه و تکثیر کتب و نشریات و آثار صوتی، ضوابط و مقررات آموزشی، پژوهشی و انصباطی ...) با این جانب رفتار خواهد شد و حق هرگونه اعتراض در خصوص احراق حقوق مکتب و تشخیص و تعیین تخلف و مجازات را از خودم سلب می‌نمایم. در ضمن، مسؤولیت هرگونه پاسخگویی به اشخاص اعم از حقیقی و حقوقی و مراجع ذی‌صلاح (اعم از اداری و قضایی) به عهده‌ی اینجانب خواهد بود و دانشگاه هیچ‌گونه مسؤولیتی در این خصوص نخواهد داشت.

نام و نام خانوادگی: هانا کریمان

امضا و تاریخ:

مجوز بهره‌برداری از پایان‌نامه

بهره‌برداری از این پایان‌نامه در چهارچوب مقررات کتابخانه و با توجه به محدودیتی که توسط استاد راهنمای شرح زیر تعیین می‌شود، بلامانع است:

- بهره‌برداری از این پایان‌نامه/رساله برای همگان بلامانع است.
- بهره‌برداری از این پایان‌نامه/رساله با اخذ مجوز از استاد راهنمای، بلامانع است.
- بهره‌برداری از این پایان‌نامه/رساله تا تاریخ ممنوع است.

نام استاد راهنمای: دکتر بهروز مینایی

تاریخ:

امضا:

تقدیم به:

پدر، مادر و برادر عزیزم.

تشکر و قدردانی:

تشکر می‌کنم از استاد دلسوزم، جناب دکتر بهروز مینایی که سال‌ها من را در حوزه نرم‌افزار و بازی‌سازی راهنمایی بودند و بدون راهنمایی‌های ایشان این پروژه پیشرفته نداشت. از تمام دوستان عزیزم تشکر می‌کنم که در این راه پر پیچ و خم همراه و همدل بودند و تمام سختی‌های راه را با من شریک شدند. از برادر عزیزم تشکر می‌کنم که همیشه برای من الگویی در مسیر زندگی، درس، کار و پیشرفت بوده و همیشه مشاور و کمک‌رسان من بوده. در نهایت تشکر می‌کنم از پدر و مادر عزیزم که بدون محبت‌ها و حمایت‌های ایشان، نه تنها هرگز در مسیر علم و دانش قرار نمی‌گرفتم بلکه سختی‌های زندگی مانع از هرگونه پیشرفت من می‌شد. همیشه و در هر لحظه‌ای دست‌بوس این دو عزیز زندگیم هستم.

چکیده

نقشه‌سازی اولیه یک مرحله بسیار اساسی در روند تولید نقشه بازی توسط یک طراح بازی^۱ است که ابتدایی‌ترین نقشه بازی را ایجاد می‌کند ، یکی از مهم ترین مراحل آن ایجاد یک نمای کلی از نقشه به صورت ساده است که به طراحان بازی کمک می‌کند تا ایده‌ها و الگوهای اصلی بازی را به شکل گرافیکی بیان کنند تبدیل نقشه سازی اولیه به یک صحنه در یونیتی ممکن است چالش‌ها و سختی‌های خود را برای یک طراح بازی داشته باشد مانند : آماده سازی فایل‌ها ، استفاده از یونیتی (ایجاد یک پروژه جدید در یونیتی، ساخت یک صحنه جدید و ...) ، ایجاد اجزا بازی (استفاده از ادیتور یونیتی برای ایجاد اجزا بازی و قرار دادن آنها در موقع مناسب بر اساس نقشه و ...) در این پایان نامه، ما بروی تولید ابزاری برای تولید صحنه‌ی سه بعدی و یا دو بعدی نقشه کشیده توسط طراح در یونیتی تمرکز می‌کنیم، با هدف ساده کردن این مسایل برای طراحان بازی و کمک به آن‌ها تا بتوانند بدون نیاز به کارکردن با عنصر‌های بخصوص یونیتی (جزء برخورد دهنده^۲ و ...) و کد نویسی ، صحنه‌ی اولیه‌ی خود را ایجاد کنند . این ابزار از پردازش تصویر استفاده می‌کند تا نقشه‌ی فیزیکی داده شده (کشیده شده بروی کاغذ) و اسکن شده را دریافت کند و اجزا نقشه را که به صورت ساده شبیه سازی شده‌اند (مانند : دایره ، مثلث ، مستطیل و ..) را تشخیص دهد و آن‌ها را به شی داده شده به برنامه تبدیل کند سپس در صحنه‌ی ایجاد شده قرار دهد. پردازش تصویر در این پروژه با استفاده از ابزار opencv و یکی از الگوریتم‌های آن به نام TM_CCOEFF_NORMED انجام شده که یکی از روش‌هایی است که برای تطبیق قالب استفاده می‌شود. تطبیق الگو تکنیکی است که در بینایی کامپیوتر برای یافتن یک تصویر فرعی (الگو) در یک تصویر استفاده می‌شود. روش شده است. با استفاده از کتابخانه tkinter، استفاده از این ابزار را برای کاربران ان ساده‌تر شده است . خروجی این برنامه فایلی دو بعدی یا سه بعدی است (انتخاب کاربر) که با قرار دادن این فایل در یونیتی و پس از انجام یک سری از پردازش‌ها به صحنه‌ی بازی مورد نظر تبدیل می‌شود.

واژه‌های کلیدی: نقشه‌سازی اولیه ، عنصر‌های بخصوص یونیتی ، پردازش تصویر ، opencv ، کتابخانه .tkinter^۳

Game designer^۱
collider component²
library³

فهرست مطالب

۱	فصل اول:
۱	مقدمه
۲	۱- مقدمه
۲	۲- تعیین محیط بازی
۳	۳- تشكیل ساختار بازی
۳	۴- قرارگیری المان‌ها و تعیین ابعاد و اندازه‌ها
۴	۵- انگیزه‌های پژوهش
۴	۶- دستاوردهای پژوهش
۴	۷- ساختار پایان نامه
۵	فصل ۲:
۵	تعاریف و مفاهیم پایه‌ای
۶	۱- مقدمه
۶	۲- رابط کاربری گرافیکی
۷	۳- پردازش تصویر
۷	۱-۳-۲- کتابخانه opencv
۸	۲-۳-۲- تطبیق چند قالب
۸	۳-۳-۲- سرکوب غیر حداکثری برای تشخیص شی
۹	۴-۳-۲- تطبیق قالب چند مقیاسی
۱۰	۱-۴-۳-۲- محدودیت‌ها و معایب
۱۲	۴-۲- کتابخانه بلندر
۱۳	۵-۲- شبکه‌های عصبی کانولوشنال
۱۴	۱-۵-۲- وزن‌ها و تعصبات مشترک
۱۴	۲-۵-۲- لایه‌های طبقه‌بندی
۱۴	۳-۵-۲- چه زمانی باید از شبکه‌های عصبی کانولوشنال استفاده کرد؟
۱۵	فصل ۳:
۱۵	مروری بر کارهای مرتبط
۱۶	۱-۳- مقدمه
۱۶	۲-۳- موبایل نت
۱۶	۱-۲-۳- پیچیدگی قابل تفکیک عمیق
۱۷	۳-۳- سیستم تشخیص شکل سه بعدی
۱۸	۱-۳-۳- بارگیری موبایل نت V1 به عنوان مدل پایه
۱۸	۲-۳-۳- بارگیری مجموعه داده
۱۹	۳-۳-۳- ساخت مدل دقیق
۲۰	۴-۳-۳- آموزش مدل دقیق با استفاده از یادگیری انتقالی
۲۱	۵-۳-۳- آزمایش مدل نهایی بر روی اشیاء واقعی
۲۲	۴-۳- انتقال سبک
۲۳	۵-۳- الگوریتم BSP
۲۷	۶-۳- نتیجه گیری

۲۸.....	فصل ۴: روش پیشنهادی
۲۸.....	روش پیشنهادی
۲۹.....	۱-۴- مقدمه
۲۹.....	۲-۴- مدل کلی روش ارائه شده
۲۹.....	۳-۴- پیاده سازی سرکوب غیر حداکثری
۳۱.....	۴-۴- پیاده سازی پردازش تصویر
۳۱.....	۱-۴-۴- کتابخانه های استفاده شده
۳۱.....	۲-۴-۴- تابع پیش پردازش
۳۲.....	۳-۴-۴- تابع تطبیق
۳۴.....	۵-۴- پیاده سازی ساخت نقشه سه بعدی
۳۵.....	۶-۴- پیاده سازی ساخت نقشه دو بعدی
۳۶.....	۷-۴- رابط کاربری
۳۷.....	۸-۴- نمونه خروجی
۳۸.....	۹-۴- نتیجه گیری
۳۹.....	فصل ۵: ارزیابی روش پیشنهادی
۳۹.....	ارزیابی روش پیشنهادی
۴۰.....	۱-۵- مقدمه
۴۰.....	۲-۵- محیط آزمایش روش ارائه شده
۴۱.....	۳-۵- ارزیابی
۴۱.....	۴-۵- نتایج مقایسه
۴۲.....	۵-۵- مقایسه ابزار OPENCV با ابزارهای دیگر
۴۳.....	۶-۵- نتیجه گیری
۴۴.....	فصل ۶: نتیجه گیری و کارهای آینده
۴۴.....	نتیجه گیری و کارهای آینده
۴۵.....	۱-۶- مقدمه
۴۵.....	۲-۶- نتیجه گیری
۴۵.....	۳-۶- کارهای آینده
۴۷.....	مراجع
۵۰.....	واژه نامه

فهرست شکل‌ها

شکل(۱-۱): نمونه‌ی محیط‌های مختلف در بازی‌ها [۱]	۲
شکل(۱-۲): نمونه‌ی یک نقشه‌ی اولیه ساخته شده [۲]	۳
شکل(۲-۱): نمونه‌ای از رابط کاربری گرافیکی با استفاده از TKINTER [۳]	۶
شکل(۲-۲): نمونه‌ای از تطبیق قالب [۴]	۷
شکل(۳-۲): سمت چپ: تصویر ورودی حاوی الگویی از «الماس» [۵]	۸
شکل(۴-۲): طبقه بندی کننده [۶]	۹
شکل(۵-۲): ب تطبیق الگویی چند مقیاسی برای یافتن الگو در تصویر [۷]	۱۰
شکل(۶-۲): وقتی اندازه تصویر الگو (سمت چپ) با اندازه منطقه در تصویر (راست) مطابقت نداشته باشد [۸]	۱۱
شکل(۷-۲): نمونه‌ی شبکه عصبی کانولوشنال [۸]	۱۳
شکل(۸-۲): مثالی از یک شبکه با چندین لایه کانولوشن.. [۸]	۱۴
شکل(۱-۳): بارگیری موبایل نت	۱۸
شکل(۲-۳): کلاس استفاده شده شامل مکعب، استوانه، کروی و کره	۱۸
شکل(۳-۳): تمامی تصاویر مورد استفاده برای آموزش و تست به شرح بالا پیش پردازش شده‌اند.	۱۹
شکل(۴-۳): تجسم نمونه‌ای از مجموعه داده‌های آموزشی. [۱۰]	۱۹
شکل(۵-۳): ساخت مدل دقیق (هر نورون مربوط به یک کلاس خروجی از اشکال است)	۲۰
شکل(۶-۳): آموزش مدل دقیق.	۲۰
شکل(۷-۳): آموزش مدل دقیق.	۲۱
شکل(۸-۳): نتایج حاصل از آزمایش. [۱۰]	۲۱
شکل(۹-۳): نمونه‌ای از انتقال سبک [۴۱]	۲۳
شکل(۱۰-۳): نمونه‌ای از جداسازی درخت چهارتایی [۲]	۲۴
شکل(۱۱-۳): یک هزار تو که با استفاده از یک درخت چهارتایی. [۱۸]	۲۵
شکل(۱۲-۳): یک هزار تو که با استفاده از یک درخت چهارتایی. [۲]	۲۵
شکل(۱۳-۳): تصویرسازی جداسازی تصادفی. [۲]	۲۶
شکل(۱۴-۳): نمونه هزار توی تصویر بالا، با استفاده از جداسازی‌ها برای موضوع‌بندی محتوای اتفاق. [۲]	۲۷
شکل(۱-۴): تصویری از رابط کاربری پیاده‌سازی شده.	۳۷
شکل(۲-۴): نمونه خروجی نقشه دو بعدی.	۳۷
شکل(۳-۴): عکس‌های داده شده به عنوان قالب‌های نقشه دو بعدی.	۳۷
شکل(۴-۴): نمونه خروجی نقشه‌ی سه بعدی.	۳۸
شکل(۵-۴): نمونه خروجی نقشه‌ی سه بعدی در محیط یونیتی.	۳۸
شکل(۵-۱): تصویری از ACTIVITY MONITOR	۴۰
شکل(۵-۲): تصویری از نقشه‌ی آزمایش شده.	۴۱

۶

فصل اول:

مقدمه

۱-۱- مقدمه

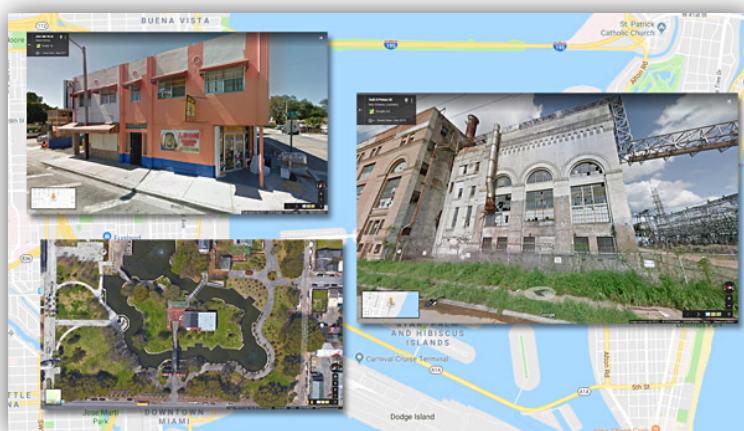
طراحی نقشه اولیه^۱ یک بازی معمولاً به عنوان یکی از مراحل مهم اولیه شناخته می‌شود و نقشه‌ها یا سطوح بازی را ایجاد می‌کند. این فرآیند از مراحل مختلفی تشکیل شده و توسط تیم طراحی بازی اجرا می‌شود. طراحی نقشه یک فرآیند خلاقانه و پیچیده است و نیاز به هماهنگی بین اعضای تیم دارد تا یک تجربه بازی جذاب و هماهنگ ارائه شود. چالش‌های طراحی نقشه اولیه بازی شامل:

۱. ایجاد ساختار بازی منطبق با داستان
۲. حفظ تنوع در محیط و المان‌های بازی
۳. انتخاب و تعیین محیط بازی مناسب
۴. مدیریت درست ابعاد و اندازه‌ها

در این پژوهش، تلاش بر ساده تر کردن و کم تر کردن این مسایل برای طراحان بازی داریم . در ادامه توضیح مختصه‌ی در باره‌ی مراحل مختلف طراحی نقشه اولیه (مرتبه با پروژه ما) داده شده است.

۱-۲- تعیین محیط بازی

محیط بازی ممکن است شهرها، جنگل‌ها، زیرزمین‌ها یا سایر مکان‌ها باشد. تعیین محیط بازی بر اساس داستان بازی و نیازهای گیم‌پلی انجام می‌شود. انتخاب محیط بازی از این جهت مهم است که محیط بازی با پیچیدگی‌ها و فضاسازی‌های مختلف میتواند در پیاده سازی و کارایی الگوریتم تاثیرگذار باشد. محیط‌هایی که شامل اشیا بیشتری می‌شوند زمان پیاده سازی آن‌ها در زمان بیشتری انجام می‌شوند و نیازمند منابع بیشتری برای پیاده سازی دارند.



[۱]: نمونه‌ی محیط‌های مختلف در بازی‌ها

۱-۳- تشکیل ساختار بازی

طراحان نقشه باید به دقت ساختار بازی را مشخص کنند. این شامل مواردی مانند مراحل مختلف، مکان‌های کلیدی، مسیرها و ارتباطات بازیکن با محیط است. قبل از شروع طراحی مراحل توسط الگوریتم، طراحان بازی باید فرضیات متفاوت را در نظر بگیرند تا از پیاده سازی اشتباه مراحل و سو مصرف منابع جلوگیری شود.

۱-۴- قرارگیری المان‌ها و تعیین ابعاد و اندازه‌ها

در این مرحله، المان‌های بازی مانند موائع، اشیاء قابل تعامل، نقاط نجات در نقشه قرار می‌گیرند. این باعث ایجاد چالش‌ها و تنوع در گیم پلی می‌شود. نقشه باید به اندازه مناسبی باشد تا گیم پلی بهینه و جذاب باشد، این شامل ابعاد نقشه و اندازه مناسب برای فعالیت‌های مختلف است.



[۲]: نمونه یک نقشه‌ی اولیه ساخته شده [۲]

۱-۵- انگیزه‌های پژوهش

همانطور که پیش‌تر توضیح داده شده طراحی ابتدایی نقشه بازی مراحل مختلف و چالش‌هایی دارد. در اکثر بازی‌ها طراحان بازی نیاز دارند تا با تنظیم متغیرهای مربوط، نقشه‌ی تولید شده‌ی اولیه‌ی متناسب با بازی را استخراج کنند. اما در بسیاری از موقع تنظیم متغیرها و کار کردن با ابزارها و برنامه‌ها، برای طراحانی که در زمینه‌های فنی تجربه‌ای ندارند کار دشواری است و همین امر زمینه‌هایی که طراحان بتوانند بدون دخالت و کمک همکاران به کار خود ادامه دهنند را محدود می‌کند. از طرفی تعامل طراحان بازی که تجربه برنامه نویسی و توسعه فنی بازی را ندارند با توسعه دهنده‌های بازی، خود چالشی است که گاه‌ها خروجی‌های دلخواهی ندارند. همچنین در برخی از بازی‌ها که شامل مراحل بسیار زیادی دارند، خودکار کردن روند پیاده سازی مراحل بازی ارزش زیادی دارد.

از این رو یافتن روشی جدید که بتواند انجام این کار را ساده‌تر و سریع‌تر کند می‌تواند حائز اهمیتی فراوان باشد.

۱-۶- دستاوردهای پژوهش

بنابراین، در پژوهه فعلی به دنبال بدست آوردن روش کاربردی تری برای تولید یک فضای نقشه و مرحله بازی هستیم که کمک می‌کند این عمل سریع‌تر، دقیق‌تر و با استفاده از منابع کمتری محقق شود. نوآوری اصلی این پژوهش نسبت به روش سنتی و پایه‌ای در این است که با استفاده از ابزارهای تحلیل تصویر و مدرن تولید این محتواها را به صورت خودکار انجام دهد.

۱-۷- ساختار پایان‌نامه

ادامه‌ی پژوهه به شش فصل تقسیم شده است. در فصل ۲ به توضیح برخی از مفاهیم پایه در این حوزه می‌پردازیم. در فصل ۳ کارهای مرتبط انجام شده در این حوزه را بررسی می‌کنیم و در فصل ۴ نحوه پیاده‌سازی روش پیشنهادی را شرح می‌دهیم. در فصل ۵ به آزمایش و شبیه سازی روش پیشنهادی می‌پردازیم. در انتهای در فصل ۶ به جمع‌بندی و نتیجه گیری از پژوهش انجام شده و به ارائه پیشنهاد برای کارهای آینده پرداخته شده است.

فصل 2

تعریف و مفاهیم پایه‌ای

۱-۲ - مقدمه

در این فصل به معرفی برخی مفاهیم پایه‌ای رابط کاربری گرافیکی^۱، پردازش تصویر^۲، و رابط برنامه نویسی کاربردی^۳ در زبان پایتون میپردازیم اهمیت این فصل در درک بهتر روند پروژه و نحوه کار آن است.

۲-۲ - رابط کاربری گرافیکی

رابط کاربری گرافیکی (UI) به نقطه تعامل بین یک کاربر و یک دستگاه دیجیتال یا برنامه نرم افزاری اشاره دارد. طراحی UI شامل ایجاد عناصر بصری، چیدمان و ظاهر و احساس کلی یک رابط دیجیتال برای اطمینان از تجربه کاربری مثبت و موثر است.

به عبارت ساده‌تر، UI شامل همه چیزهایی است که کاربر می‌تواند در یک پلتفرم دیجیتال با آن تعامل داشته باشد، از جمله دکمه‌ها، نمادها، منوها، متن، تصاویر و موارد دیگر. هدف از طراحی UI این است که تعامل بین کاربر و سیستم را تا حد امکان ساده، کارآمد و لذت بخش کند. در پایتون، Tkinter یک کتابخانه محبوب برای ساخت رابط‌های گرافیکی کاربر است.

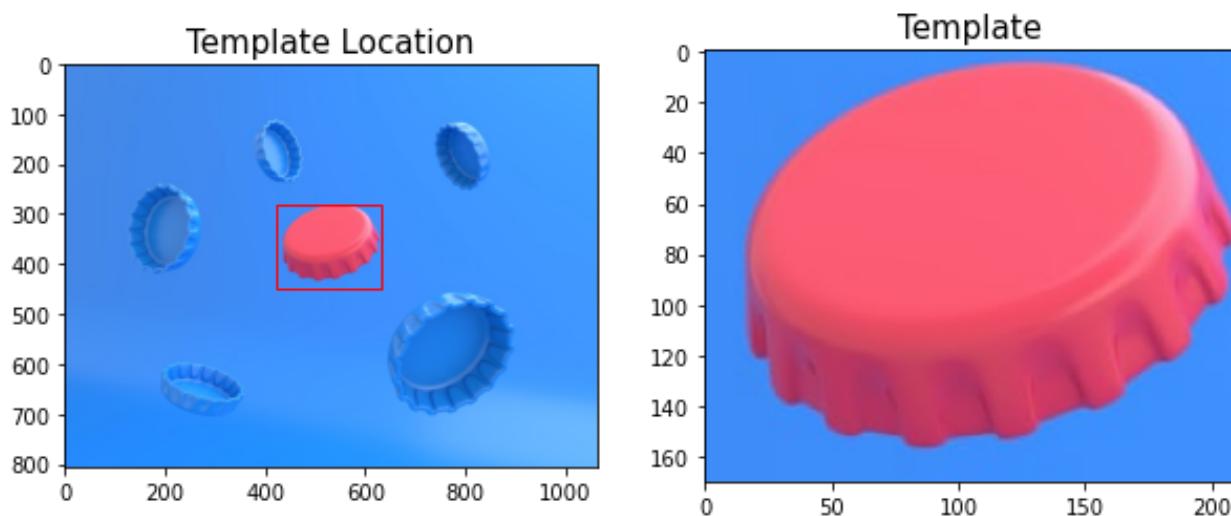
ایجاد یک رابط کاربری در پایتون با Tkinter یک فرآیند ساده است. با درک اصول اولیه ویجت‌ها، مدیریت چیدمان، مدیریت رویداد و ویژگی‌های اضافی، می‌توانید برنامه‌های تعاملی و کاربرپسند طراحی کنید. با مراجعه به اسناد رسمی و آزمایش با اجزای مختلف رابط کاربری و عملکردهای مختلف، راحت‌تر به بررسی Tkinter بپردازید.



شکل(۱-۲): نمونه‌ای از رابط کاربری گرافیکی با استفاده از Tkinter^[۲]

۳-۲- پردازش تصویر

در چشم انداز وسیع بینایی کامپیوتر، دو جزء کلیدی در استخراج اطلاعات معنی دار از داده های بصری نقش اساسی دارند: پردازش تصویر و تشخیص اشیا^۱. این فناوری ها صنایع متعددی را متحول کردند، از مراقبت های بهداشتی و خودرو گرفته تا امنیت و سرگرمی. پردازش تصویر یک مفهوم اساسی در بینایی کامپیوتراست که شامل دستکاری و بهبود داده های بصری برای استخراج اطلاعات مربوطه است. این می تواند طیف گسترده ای از وظایف، مانند فیلتر کردن تصویر، بخش بندی، و استخراج ویژگی را در بر بگیرد. هدف اصلی پردازش تصویر، بهبود کیفیت تصاویر است و آنها را برای تجزیه و تحلیل بعدی مناسب تر می کند[۴].



شکل(۲-۲): نمونه های از تطبیق قالب [۴]

۱-۳-۲- کتابخانه opencv^۲

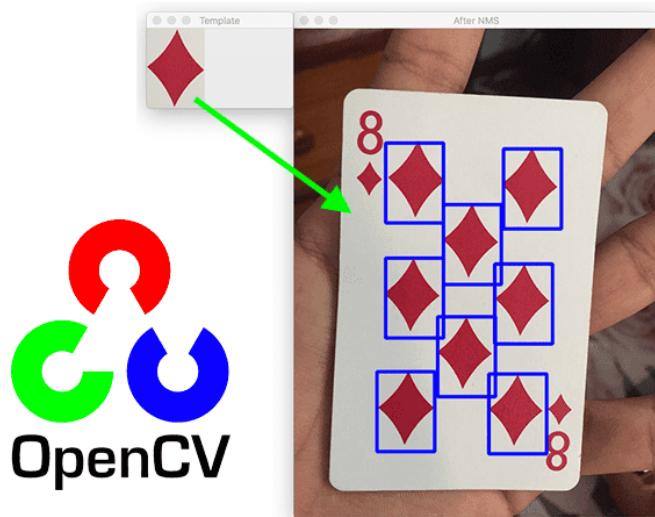
یک کتابخانه بینایی و پردازش تصویر کامپیوترا منبع باز است که مجموعه وسیعی از ابزارها و عملکردها را برای تجزیه و تحلیل، دستکاری و درک داده های بصری ارائه می دهد. OpenCV که در C++^۲ توسعه یافته و در پایتون، جاوا و سایر زبان ها موجود است، به طور گسترده در زمینه های مختلف از جمله بینایی کامپیوترا، یادگیری ماشین، رباتیک و غیره استفاده می شود.[۱۹]

Object detection^۱

Open Source Computer Vision Library²

۲-۳-۲- تطبیق چند قالب^۱

تطبیق چند قالب یک تکنیک بینایی کامپیوتراست که برای تشخیص چندین نمونه از یک الگوی مشخص (یک تصویر یا الگوی کوچک) در یک تصویر بزرگتر استفاده می‌شود. به جای جستجوی تنها یک الگو، این روش شامل تطبیق چندین قالب به طور همزمان است. این رویکرد زمانی سودمند است که تغییرات ظاهری یا مقیاس در بین نمونه‌های شی شناسایی شده وجود داشته باشد. تطبیق چند قالب معمولاً در تشخیص اشیا، تجزیه و تحلیل تصویر و برنامه‌های بینایی رایانه‌ای استفاده می‌شود که در آن مستحکم بودن در ظاهر شیء متنوع مورد نیاز است [۵].



شکل(۲-۲): سمت چپ: تصویر ورودی حاوی الگویی از «الماس» است. وسط: تصویر الگوی نماد «الماس» که ما می خواهیم شناسایی کنیم. راست: خروجی اعمال تطابق اولیه قالب با OpenCV تطبیق الگوی استاندارد برای کنترل چندین تشخیص طراحی نشده است [۵].

۲-۳-۳- سرکوب غیر حداقلی برای تشخیص شی^۲

سرکوب غیر حداقلی (NMS) تکنیکی است که معمولاً در تشخیص اشیا برای حذف جعبه‌های مرزی ^۳ اضافی یا همپوشانی استفاده می‌شود و فقط مطمئن‌ترین و دقیق‌ترین پیش‌بینی‌ها را حفظ می‌کند. به طور خلاصه: خروجی تشخیص: الگوریتم‌های تشخیص شی اغلب پیش‌بینی‌های جعبه مرزی متعددی را برای یک شی در یک تصویر ایجاد می‌کنند که منجر به افزونگی می‌شود.

تشخیص امتیاز: هر جعبه مرزی با یک امتیاز اطمینان همراه است که نشان دهنده اطمینان الگوریتم در صحت پیش‌بینی است.

فرآیند سرکوب: سرکوب غیر حداقلی شامل مرتب‌سازی جعبه‌های مرزی بر اساس امتیازات اطمینان آنها به ترتیب نزولی است. با شروع با جعبه با بالاترین امتیاز، این جعبه را به عنوان یک تشخیص نگه می‌دارد و هر جعبه

Multi template matching¹

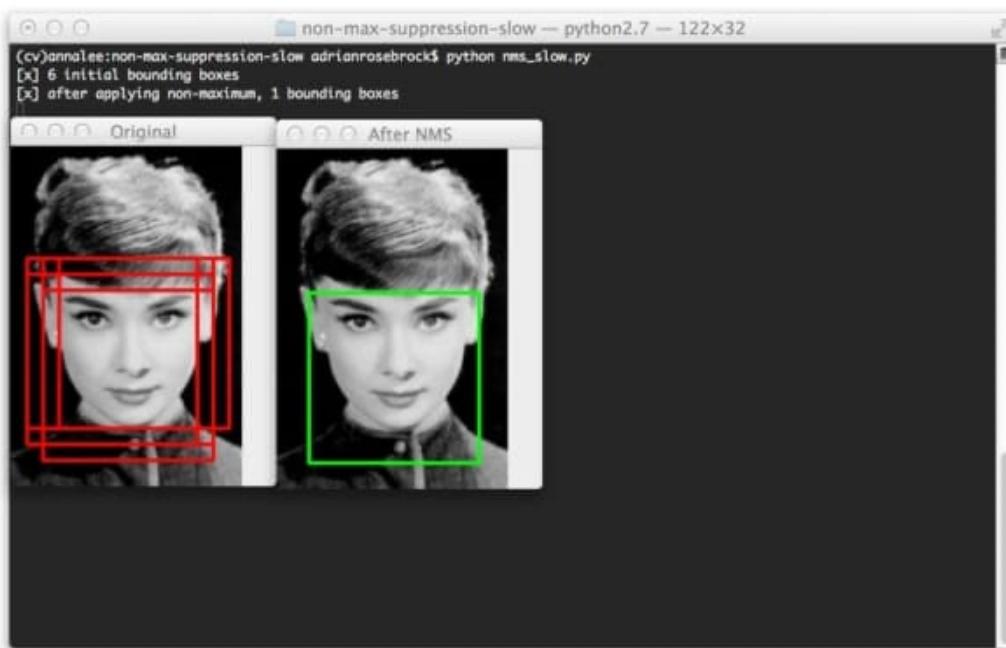
Non-maximum suppression²

Bounding box³

دیگری را که دارای همپوشانی قابل با کادر انتخاب شده باشد سرکوب می کند (حذف می کند).

فرآیند تکراری: این فرآیند تا زمانی که همه جعبه‌های مرزی در نظر گرفته شوند، تکرار می‌شود، که منجر به مجموعه نهایی تشخیص‌های غیر همپوشانی و با اطمینان بالا می‌شود.

با اعمال NMS ، الگوریتم تضمین می‌کند که فقط مطمئن‌ترین و غیر همپوشانی‌ترین جعبه‌های مرزی حفظ می‌شوند، که افزونگی را کاهش می‌دهد و دقت نتایج تشخیص شی را بهبود می‌بخشد.[۶]



شکل (۲-۴): طبقه بندی کننده ما در ابتدا شش جعبه محدود کننده را شناسایی کرد، اما با اعمال مهار غیر حداقل، تنها یک جعبه (درست) محدود کننده باقی می‌ماند[۶].

۴-۳-۲- تطبیق قالب چند مقیاسی^۱

تطبیق قالب چند مقیاسی، توسعه‌ای از تکنیک تطبیق الگوی سنتی است که برای بهبود استحکام تشخیص شی در مقیاس‌های مختلف طراحی شده است.

تطبیق الگو در مقیاس‌های چندگانه: به جای استفاده از یک الگو برای تطبیق، تطبیق الگوی^۲ چند مقیاسی شامل اعمال الگوریتم تطبیق الگو در مقیاس‌های مختلف تصویر ورودی است.

نمایش فضای مقیاس: تصویر اغلب به یک فضای مقیاس تبدیل می‌شود، جایی که همان الگو در سطوح مختلف وضوح تصویر مطابقت داده می‌شود. این به تشخیص اجسام در اندازه‌های مختلف کمک می‌کند. رویکرد هرمی^۳ : تطبیق الگوی چند مقیاسی را می‌توان با استفاده از هرم‌های تصویر پیاده سازی کرد، که در آن

Multi-scale template matching¹

Template matching²

Pyramid approach³

تصویر به طور متوالی نمونه برداری می شود تا یک هرم از تصاویر ایجاد شود. سپس تطبیق الگو در هر سطح از هرم انجام می شود.

استحکام در تغییرات مقیاس: تطبیق الگوی چند مقیاسی به ویژه زمانی مفید است که اندازه شی مورد نظر در تصویر ناشناخته یا متفاوت باشد. این به الگوریتم اجازه می دهد تا شی را در طیف وسیعی از مقیاس ها شناسایی کند و عملکرد تشخیص را بهبود بخشد.

با در نظر گرفتن مقیاس های متعدد در طول تطبیق الگو، این تکنیک توانایی مکان یابی اشیاء را در تصاویری که اندازه شی ممکن است متفاوت باشد یا نامشخص باشد، افزایش می دهد و در سناریوهایی با تغییرات مقیاس قوی تر می شود.^[7]



شكل(۵-۲): با موفقیت تطبیق الگوی چند مقیاسی برای یافتن الگو در تصویر اعمال شد^[7]

۱-۴-۳-۲- محدودیت ها و معایب

البته، استفاده از تطبیق قالب ساده، حتی تطبیق قالب چند مقیاسی، محدودیت ها و اشکالات قابل توجهی دارد. در حالی که می توانیم تغییرات در ترجمه و مقیاس‌بندی را مدیریت کنیم، رویکرد ما نسبت به تغییرات چرخش یا تبدیل های غیر وابسته قوی نخواهد بود.

اگر نگران چرخش در تبدیل‌های غیر وابسته هستیم، بهتر است برای شناسایی نقاط کلیدی، استخراج توصیفگرهای ثابت محلی و اعمال تطابق نقاط کلیدی وقت بگذاریم.

اما در موردی که الگوهای ما نسبتاً سفت و سخت هستند و از طریق یک نقشه لبه به خوبی تعریف می‌شوند و ما فقط به ترجمه و مقیاس‌بندی اهمیت می‌دهیم، تطبیق قالب در مقیاس چندگانه می‌تواند نتایج بسیار خوبی را با تلاش کم به ما ارائه دهد.

در نهایت، مهم است که به خاطر داشته باشید که تطبیق الگو کار خوبی برای تشخیص اینکه آیا یک شی در یک تصویر ظاهر نمی‌شود، انجام نمی‌دهد. مطمئن‌آمیزی آستانه‌هایی را روی ضریب همبستگی تعیین کنیم، اما در عمل این قابل اعتماد و قوی نیست. اگر به دنبال یک رویکرد قوی‌تر هستید، باید تطبیق نقاط کلیدی را بررسی کنید.



شکل (۶-۲): وقتی اندازه تصویر الگو (سمت چپ) با اندازه منطقه در تصویر (راست) مطابقت نداشته باشد [۸]

۴-۲- کتابخانه بلندر^۱ :

بلندر یک نرم افزار گرافیکی کامپیوترا سه بعدی منبع باز^۲ قدرتمند است که به کاربران اجازه می دهد فیلم های متحرک، جلوه های بصری، هنری، بازی های سه بعدی و غیره بسازند. در حالی که بلندر خود یک کتابخانه پایتون نیست، یک رابط برنامه نویسی کاربردی جامع پایتون را ارائه می دهد که به توسعه دهنده‌گان اجازه می دهد تا با عملکردهای بلندر به صورت برنامه نویسی تعامل داشته باشند و آنها را دستکاری کنند. در اینجا چند نکته کلیدی در مورد رابط برنامه نویسی کاربردی بلندر برای پایتون^۳ آورده شده است:

اسکریپت‌نویسی و اتوماسیون: برنامه نویسی کاربردی بلندر برای پایتون به کاربران امکان می دهد تا کارهای مختلف را اسکریپت کنند و گرددش‌های کاری را در محیط بلندر خود کار کنند. این به ویژه برای کارهای تکراری، پردازش دسته‌ای یا ایجاد ابزارهای سفارشی مفید است.

دسترسی به ویژگی‌های بلندر: برنامه نویسی کاربردی دسترسی به طیف گسترده‌ای از ویژگی‌های بلندر، از جمله دستکاری مش، انیمیشن، نورپردازی، مواد، رندر و غیره را فراهم می کند. این به توسعه دهنده‌گان اجازه می دهد تا قابلیت‌های بلندر را بر اساس نیازهای خاص خود سفارشی کرده و گسترش دهند.

رابط اسکریپت API: پایتون بلندر از یک رابط برنامه نویسی استفاده می کند که به کاربران اجازه می دهد اسکریپت‌های پایتون را مستقیماً در محیط بلندر بنویسند. این اسکریپت‌ها را می توان برای انجام اقدامات یا عملیات خاص اجرا کرد.

مستندات گسترده: رابط برنامه نویسی کاربردی بلندر برای پایتون به خوبی مستند شده است و آن را برای توسعه دهنده‌گانی که می خواهند قابلیت‌های آن را کاوش کرده و استفاده کنند قابل دسترسی است. این اسناد شامل مثال‌ها، مواد مرجع و راهنمایی برای جنبه‌های مختلف API است.

توسعه افزونه‌ها: بلندر از توسعه افزونه‌ها با استفاده از پایتون پشتیبانی می کند. افزونه‌ها افزونه‌های سفارشی هستند که می توانند به راحتی در بلندر ادغام شوند تا عملکرد آن را افزایش دهند. بسیاری از افزونه‌ها در پایتون نوشته شده اند و آن را به زبانی همه کاره برای گسترش قابلیت‌های بلندر تبدیل می کند.

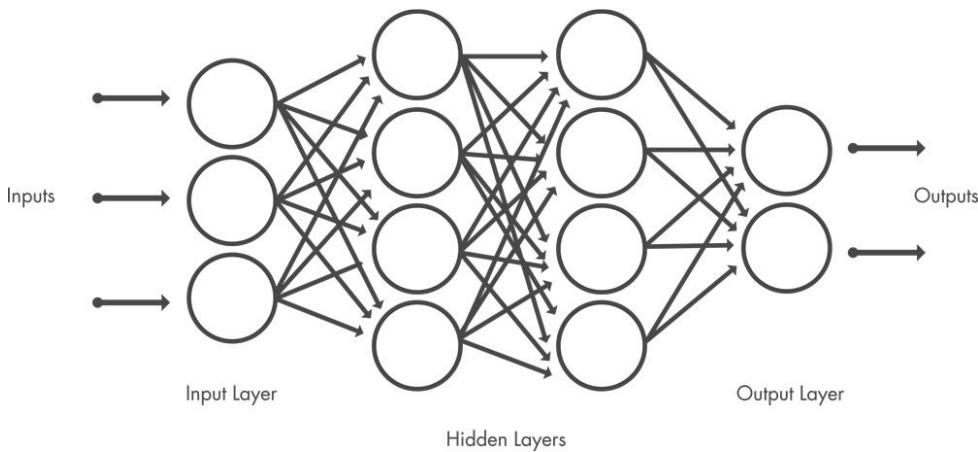
Blender^۱

Open source^۲

Api blender python^۳

۱-۵-۲- شبکه های عصبی کانولوشنال^۱

یک شبکه عصبی کانولوشنال می تواند دهها یا صدها لایه داشته باشد که هر کدام تشخیص ویژگی های مختلف یک تصویر را یاد می گیرند. فیلترها بر روی هر تصویر آموزشی با وضوح های مختلف اعمال می شوند و خروجی هر تصویر پیچیده به عنوان ورودی لایه بعدی استفاده می شود. فیلترها می توانند به عنوان ویژگی های بسیار ساده مانند روشنایی و لبه ها شروع شوند و پیچیدگی را تا ویژگی هایی افزایش دهند که به طور منحصر به فرد شی را تعریف می کنند. یک CNN از یک لایه ورودی، یک لایه خروجی و بسیاری از لایه های پنهان در بین آنها تشکیل شده است.



شکل(۷-۲): نمونهی شبکه عصبی کانولوشنال.[۸]

این لایه ها عملیاتی را انجام می دهند که داده ها را با هدف یادگیری ویژگی های خاص داده ها تغییر می دهد.

سه لایه از رایج ترین لایه ها عبارتند از پیچیدگی^۲، یا واحد خطی اصلاح شده^۳ و ادغام^۴.

- پیچیدگی تصاویر ورودی را از طریق مجموعه ای از فیلترهای کانولوشن قرار می دهد که هر کدام ویژگی های خاصی را از تصاویر فعال می کنند.
- واحد خطی اصلاح شده با نگاشت مقادیر منفی به صفر و حفظ مقادیر مثبت، امکان آموزش سریعتر و مؤثرتر را فراهم می کند. گاهی اوقات از آن به عنوان فعال سازی یاد می شود، زیرا فقط ویژگی های فعال شده به لایه بعدی منتقل می شوند.
- ادغام خروجی را با انجام نمونه برداری پایین غیر خطی ساده می کند و تعداد پارامترهایی را که شبکه باید یاد بگیرد کاهش می دهد.

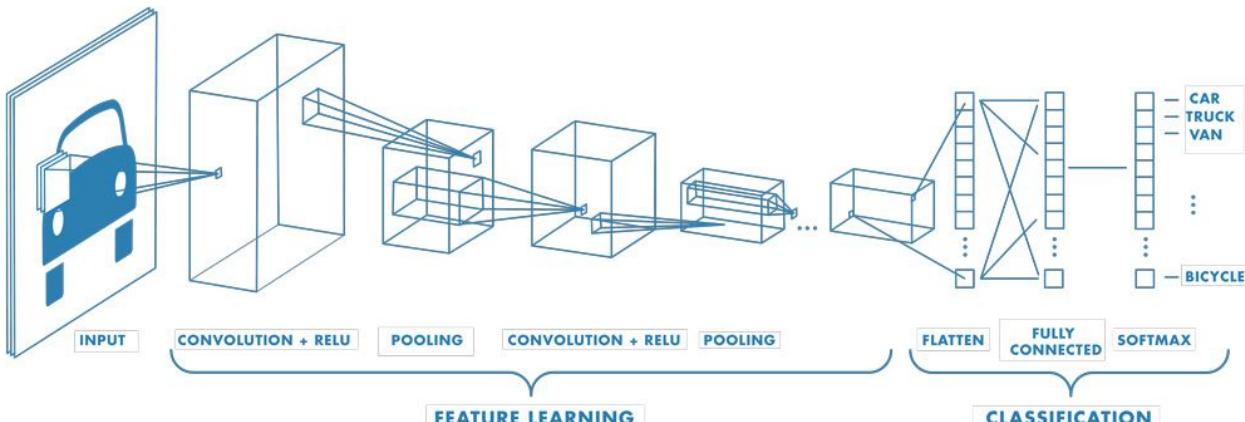
این عملیات در ده ها یا صدها لایه تکرار می شود و هر لایه شناسایی ویژگی های مختلف را یاد می گیرد.

(CNN) Convolutional neural network^۱

convolution^۲

Rectified linear unit (ReLU)^۳

Pooling^۴



شکل(۲-۸): مثالی از یک شبکه با چندین لایه کانولوشن. فیلترها برای هر تصویر آموزشی با رزولوشن های مختلف اعمال می شوند و خروجی هر تصویر پیچیده به عنوان ورودی لایه بعدی استفاده می شود.[۸]

شکل(۲-۸): مثالی از یک شبکه با چندین لایه کانولوشن. فیلترها برای هر تصویر آموزشی با رزولوشن های مختلف اعمال می شوند و خروجی هر تصویر پیچیده به عنوان ورودی لایه بعدی استفاده می شود.[۸]

۲-۵-۱- وزن ها و تعصبات^۱ مشترک

برخلاف یک شبکه عصبی سنتی، یک شبکه های عصبی کانولوشنال دارای وزن ها و مقادیر بایاس مشترک است که برای همه نورون های پنهان در یک لایه مشخص یکسان است. این بدان معناست که همه نورون های پنهان در حال شناسایی یک ویژگی، مانند لبه یا حباب، در مناطق مختلف تصویر هستند. این باعث می شود شبکه نسبت به ترجمه اشیاء در یک تصویر متتحمل شود. برای مثال، شبکه ای که برای تشخیص خودروها آموزش دیده است، می تواند این کار را در هر جایی که خودرو در تصویر باشد انجام دهد.

۲-۵-۲- لایه های طبقه بندی^۲

پس از یادگیری ویژگی ها در بسیاری از لایه ها، معماری یک شبکه های عصبی کانولوشنال به طبقه بندی تغییر می کند. لایه بعدی تا آخر یک لایه کاملاً متصل است که یک بردار با ابعاد K را خروجی می دهد (که در آن K تعداد کلاس هایی است که می توان پیش بینی کرد) و شامل احتمالات برای هر کلاس از یک تصویر طبقه بندی شده است. لایه نهایی معماری شبکه های عصبی کانولوشنال از یک لایه طبقه بندی برای ارائه خروجی طبقه بندی نهایی استفاده می کند.

۲-۵-۳- چه زمانی باید از شبکه های عصبی کانولوشنال استفاده کرد؟

وقتی حجم زیادی از داده های پیچیده (مانند داده های تصویری) دارید، از شبکه های عصبی کانولوشنال ها استفاده کنید. همچنین می توانید از شبکه های عصبی کانولوشنال با سیگنال یا داده های سری زمانی در هنگام پیش پردازش برای کار با ساختار شبکه استفاده کنید.

Biases^۱

Classification layers²

فصل 3:

مروری بر کارهای مرتبط

طی سال‌ها پژوهش در حوزه پردازش تصویر و تولید نقشه روش‌های مختلفی در زیرمجموعه این الگوریتم‌ها برای تولید نقشه ارائه شده است. اکثر این روش‌ها برای تولید هزارتو یا بازی‌های پلتفورمی کاربرد دارند. در ادامه به چند مورد از این الگوریتم‌ها میپردازیم. ابتدا به بررسی چند مورد از کارهای انجام شده در حوزه پردازش تصویر و سپس به بررسی چند روش ساخت نقشه میپردازیم که ترکیب این دو حوزه هدفی است که ما در این پژوهش آن را دنبال میکنیم.

۲-۳ - موبایل نت^۱

موبایل نت اولین مدل بینایی رایانه همراه TensorFlow است. از پیچیدگی‌های قابل تفکیک در عمق استفاده می‌کند تا تعداد پارامترها را در مقایسه با سایر شبکه‌ها با پیچیدگی‌های منظم و عمق یکسان در شبکه‌ها به میزان قابل توجهی کاهش دهد. این منجر به شبکه‌های عصبی عمیق سبک وزن می‌شود. یک پیچیدگی قابل تفکیک عمیق از دو عملیات ساخته شده است :

۱. پیچیدگی عمیق
۲. پیچیدگی نقطه‌ای

موبایل نت یک کلاس از شبکه‌های عصبی کانولوشن است که توسط گوگل منبع باز بود، و بنابراین، یک نقطه شروع عالی برای آموزش طبقه‌بندی کننده‌هایی است که بسیار کوچک و بسیار سریع هستند. سرعت و توان مصرفی شبکه متناسب با تعداد انباشته‌های چند برابر^۲ است که اندازه‌گیری تعداد عملیات ضرب و جمع ذوب شده است.^[۹]

۲-۴-۱ - پیچیدگی قابل تفکیک عمیق

این پیچیدگی از این ایده سرچشمه می‌گیرد که عمق و بعد فضایی فیلتر را می‌توان از هم جدا کرد، بنابراین نام آن قابل تفکیک است. بیایید فیلتر Sobel را مثال بزنیم که در پردازش تصویر برای تشخیص لبه‌ها استفاده می‌شود. می‌توانید ابعاد ارتفاع و عرض این فیلترها را از هم جدا کنید. فیلتر G_x را می‌توان به عنوان حاصلضرب ماتریسی $[1 \ 2 \ 1] \times [1 \ 0 \ 1]$ با $[1 \ 0 \ 1]$ - جابجا کرد.

متوجه خواهید شد که فیلتر خود را پنهان کرده است. نشان می‌دهد که نه پارامتر دارد، اما شش پارامتر دارد. این امر به دلیل جدا شدن ابعاد ارتفاع و عرض آن امکان پذیر است.

همین ایده در مورد بعد عمق مجزا از افقی (عرض*ارتفاع) صدق می‌کند، که به ما یک پیچیدگی قابل تفکیک در عمق را می‌دهد که در آن کانولوشن را به صورت عمقی انجام می‌دهیم. پس از آن از فیلتر 1×1 برای پوشش بعد عمق استفاده می‌کنیم.

Mobile net^۱

(MAC) Multiply-accumulates^۲

نکته ای که باید به آن توجه کرد این است که این کانولوشن چقدر پارامترها را کاهش می دهد تا تعداد کanal های یکسانی تولید شود. برای تولید یک کanal، به پارامترهای $3 \times 3 \times 3$ برای انجام کانولوشن از نظر عمق و پارامترهای 1×1 برای انجام بعد کانولوشن بیشتر در عمق نیاز داریم.

اما اگر به سه کanal خروجی نیاز داشته باشیم، فقط به فیلتر عمق $3 \times 3 \times 1$ نیاز داریم که در مجموع 36 پارامتر ($= 9 \times 3 \times 3 \times 3$) به ما می دهد، در حالی که برای همین تعداد کanal خروجی در کانولوشن معمولی، ما باید به فیلترهای $3 \times 3 \times 3$ نیاز داریم که در مجموع 81 پارامتر را به ما می دهد.

پیچیدگی قابل تفکیک عمقی یک پیچیدگی عمقی است که به دنبال آن یک پیچیدگی نقطه‌ای به شرح زیر است:

۱. کانولوشن عمقی^۱، پیچیدگی فضایی $DK \times DK$ از نظر کanal است. فرض کنید پنج کanal داریم، سپس پنج

کانولوشن فضایی $DK \times DK$ خواهیم داشت.

۲. کانولوشن نقطه‌ای^۲، پیچیدگی 1×1 برای تغییر بعد است.

یک نقشه از یک پیچیدگی در هر کanal ورودی به طور جداگانه است. بنابراین تعداد کanal های

خروجی آن با تعداد کanal های ورودی یکسان است. هزینه محاسباتی آن عبارت است از: $Df^2 * M * Dk^2$.

پیچیدگی نقطه‌ای یک کانولوشن با اندازه هسته 1×1 است که به سادگی ویژگی‌های ایجاد شده توسط کانولوشن

عمقی را ترکیب می‌کند. هزینه محاسباتی آن عبارت است از: $M * N * Df^2$.

۳-۳- سیستم تشخیص شکل سه بعدی

تکنیک های تشخیص شکل جنبه مهمی از بینایی کامپیوترا هستند و برای تبدیل داده های تصویر خام به نمایش های نمادین مورد نیاز برای تشخیص و مکان یابی اشیا استفاده می شوند. در این مقاله، یک نوت بوک ارائه شده است که شامل توسعه سیستمی است که چهار نوع شکل سه بعدی - مکعب، استوانه، کروی و کره را تشخیص می دهد. مدل مورد استفاده بر روی موبایل نت^۱ ساخته شده است و از مزایای یادگیری انتقال به منظور ساخت یک مدل سبک وزن اما دقیق شبکه های عصبی کانولوشنال استفاده می کند. پیاده‌سازی آن بر روی پلتفرم Cainvas انجام می‌شود، که اجرای یکپارچه نوت‌بوک‌های پایتون را برای ساختن سیستم‌های هوش مصنوعی فراهم می‌کند که در نهایت می‌توانند روی لبه مستقر شوند (یعنی یک سیستم جاسازی شده مانند [MCU های فشرده]).^۲

Depthwise convolution^۱

Pointwise convolution²

۱-۳-۳- بارگیری موبایل نت v1 به عنوان مدل پایه

موبایل نت ها بر اساس یک معماری کارآمد است که از کالولوشن های قابل تفکیک عمیق برای ساخت شبکه های عصبی عمیق سبک استفاده می کند.

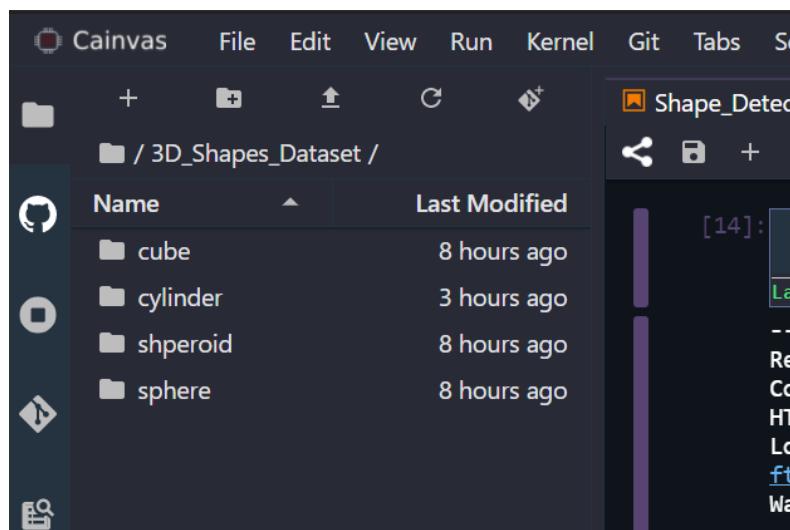
هدف از استفاده از موبایل نت برای این مورد است که این پروژه در نظر گرفته شده است تا بر روی دستگاه های تلفن همراه در لبه مستقر شود، از این رو ساخت یک مدل بر اساس یک کلاس از مدل های کارآمد (موبایل نت ها) که از قبل آموزش داده شده اند کاملا منطقی است. مجموعه‌ای از مدل های DNN تنظیم شده دقیق برای برنامه های تلفن همراه و دید تعییه شده.

```
1 base_model=MobileNet(input_shape=(IMAGE_SIZE, IMAGE_SIZE,3), alpha = ALPHA,
2                         depth_multiplier = 1, dropout = 0.001, include_top = False,
3                         weights = "imagenet", classes = 4, backend=keras.backend,
4                         layers=keras.layers,models=keras.models,utils=keras.utils)
```

شکل(۱-۳): EPOCHS=۲۰ ، ALPHA = ۰.۷۵ ، IMAGE_SIZE = ۲۲۴:۲۲۴

۲-۳-۳- بارگیری مجموعه داده

مجموعه داده مورد استفاده در اینجا یک مجموعه داده سفارشی استخراج شده با تصاویر به اندازه (۲۲۴,۲۲۴) است. این شامل ۴ دایرکتوری است که حاوی تصاویر مربوط به ۴ کلاس شکل است.



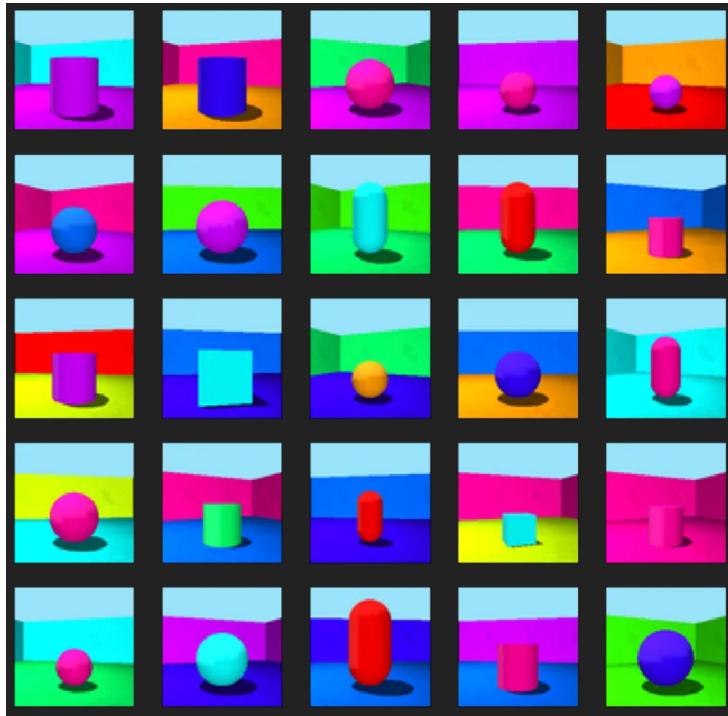
شکل(۲-۳): ۴ کلاس استفاده شده شامل مکعب، استوانه، کروی و کره

```

1 def prepare_image(file):
2     img = image.load_img(img_path + file, target_size=(IMAGE_SIZE, IMAGE_SIZE))
3     img_array = image.img_to_array(img)
4     img_array_expanded_dims = np.expand_dims(img_array, axis=0)
5     return keras.applications.mobilenet.preprocess_input(img_array_expanded_dims)

```

شکل(۳-۳): تمامی تصاویر مورد استفاده برای آموزش و تست به شرح بالا پیش پردازش شده‌اند.



شکل(۴-۳): تجسم نمونه ای از مجموعه داده های آموزشی.[۱۰].

۳-۳-۳- ساخت مدل دقیق

مدلی که باید تنظیم شود با افزودن چند لایه اضافی به مدل پایه موبایل نت ساخته شده است. در اینجا، ما ۲ لایه متراکم کاملاً متصل از ۱۰۰ و ۵۰ نورون را به ترتیب با تابع فعال سازی 'relu' و افت ۰.۵ به آخرین لایه شبکه موبایل اضافه می کنیم و یک لایه خروجی نهایی برای پیش بینی ها - که یک لایه متراکم دیگر با ۴ نورون خروجی و عملکرد فعال سازی "سافت مکس" ^۱.

```

1  def build_finetune_model(base_model, dropout, fc_layers, num_classes):
2      for layer in base_model.layers:
3          layer.trainable = False
4
5      x = base_model.output
6      x = GlobalAveragePooling2D()(x)
7
8      for fc in fc_layers:
9          # New FC layer, random init
10         x = Dense(fc, activation='relu')(x)
11         x = Dropout(dropout)(x)
12
13     # New softmax layer
14     predictions = Dense(num_classes, activation='softmax')(x)
15
16     finetune_model = Model(inputs=base_model.input, outputs=predictions)
17
18     return finetune_model
19
20 FC_LAYERS = [100, 50]
21 dropout = 0.5
22
23 finetune_model = build_finetune_model(base_model,
24                                         dropout=dropout,
25                                         fc_layers=FC_LAYERS,
26                                         num_classes=4)

```

شکل(۳-۵): ساخت مدل دقیق(هر نورون مربوط به یک کلاس خروجی از اشکال است).

۴-۳-۴-آموزش مدل دقیق با استفاده از یادگیری انتقالی

اکنون که مدل یادگیری انتقال ما ساخته شده است، می‌توانیم آن را بر روی مجموعه داده‌ای که قبلاً ذکر شد، با استفاده از تولید کننده داده‌های تصویری کراس^۱ آموزش (کوک کردن) دقیق‌تر انجام دهیم تا تصاویر را حتی بیشتر از قبل پردازش کنیم تا برای مدل شبکه تلفن همراه ما مناسب باشند، در نتیجه یک ژنراتور آموزشی تولید کنیم. (کد زیر نشان داده شده است)

```

1 train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)
2
3 train_generator=train_datagen.flow_from_directory('3D_Shapes_Dataset',
4                                                 target_size=(IMAGE_SIZE, IMAGE_SIZE),
5                                                 color_mode='rgb',
6                                                 batch_size=32,
7                                                 class_mode='categorical', shuffle=True)

```

شکل(۶-۳): آموزش مدل دقیق.

مدل CNN که قبلا ساخته شده بود، اکنون با یک بهینه ساز آدام گرداوری شده است، تلفات متقطع طبقه بندی و متریک در نظر گرفته شده در حالی که آموزش دقت مدل است. سپس مولد آموزشی تعریف شده در مدل کامپایل شده مطابق با کد زیر قرار می گیرد.

```

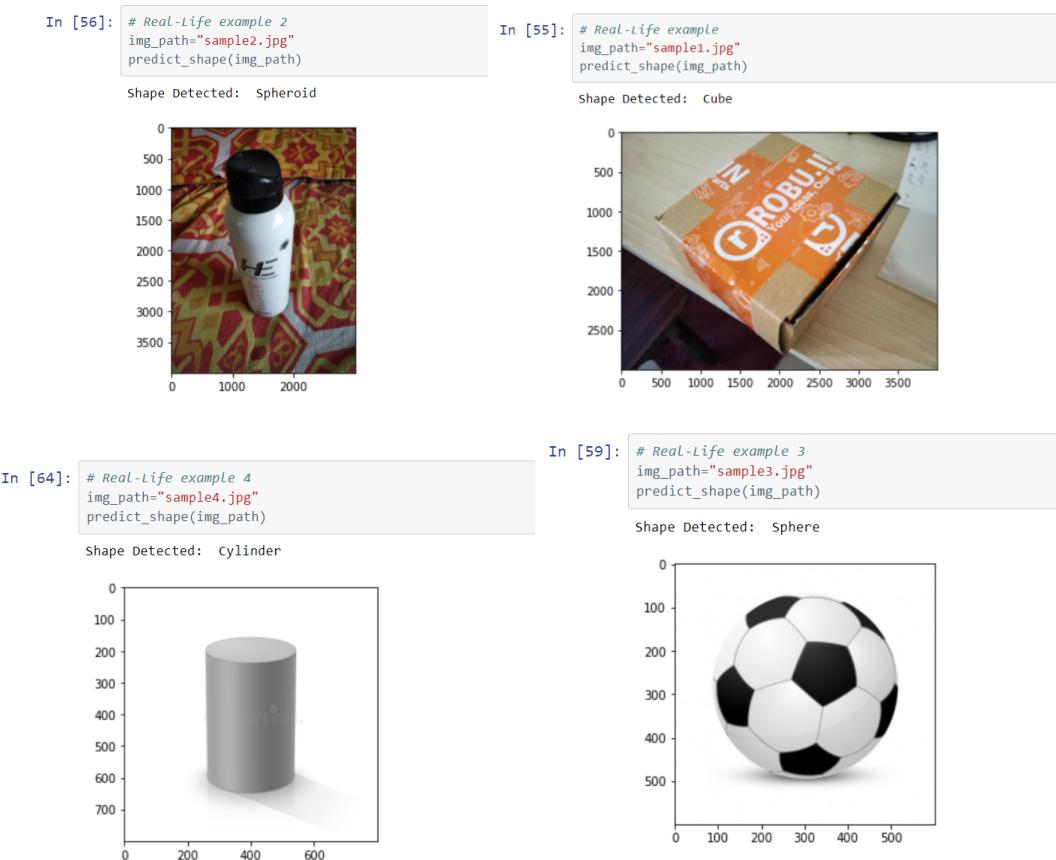
1 finetune_model.summary()
2 finetune_model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])
3 step_size_train=train_generator.n//train_generator.batch_size
4 history = finetune_model.fit_generator(generator=train_generator, steps_per_epoch=step_size_train)
5
6 finetune_model.save('shape_model.h5')

```

شکل(۷-۳): آموزش مدل دقیق.

۳-۵-۴-آزمایش مدل نهایی بر روی اشیاء واقعی

این مدل به دقت ۹۹٪ دست می یابد و از آنجایی که طبقه بندی آن فقط به شکل هندسی جسم است، حتی در چنین سطوح دقت بالایی نیز بیش از حد مناسب نیست. این مدل بر روی اشیاء دنیای واقعی و همچنین تصاویر اینترنتی برای درک بهتر قابلیت های آن آزمایش شده است.



شکل(۸-۳): نتایج حاصل از آزمایش.[۱۰]

۴-۳- انتقال سبک^۱

انتقال سبک تکنیکی در بینایی کامپیوتری و پردازش تصویر است که هدف آن اعمال سبک هنری یک تصویر بر محتوا تصویر دیگر و ایجاد یک تصویر ترکیبی جدید است. این فرآیند اغلب با استفاده از شبکه‌های عصبی عمیق، به‌ویژه شبکه‌های عصبی کانولوشنال به دست می‌آید. در اینجا یک نمای کلی از نحوه عملکرد انتقال سبک آورده شده است:

تصویر محتوا:

این تصویری است که می‌خواهید محتوا آن را حفظ کنید. این می‌تواند یک عکس یا هر تصویر دیگری باشد که می‌خواهید با یک سبک هنری خاص تغییر دهید.

تصویر سبک:

این تصویری است که می‌خواهید سبک هنری آن را در تصویر محتوا اعمال کنید. این می‌تواند یک نقاشی، اثر هنری یا هر تصویری با ویژگی‌های بصری متمایز باشد.

معماری شبکه‌های عصبی:

یک شبکه‌های عصبی کانولوشنال به دو بخش تقسیم می‌شود: لایه‌های اولیه ویژگی‌ها و جزئیات سطح پایین را ثبت می‌کنند، در حالی که لایه‌های بعدی ویژگی‌های سطح بالا و نمایش‌های انتزاعی را ثبت می‌کنند.

توابع از دست دادن:

کلید انتقال سبک، تعریف توابع از دست دادن است که تفاوت بین تصویر تولید شده و تصویر محتوا را از نظر محتوا، و بین تصویر تولید شده و تصویر سبک را از نظر سبک اندازه گیری می‌کند. این توابع از دست دادن معمولاً عبارتند از:

از دست دادن محتوا^۲: تفاوت در ویژگی‌های محتوا بین تصویر تولید شده و تصویر محتوا را اندازه گیری می‌کند.

از دست دادن سبک^۳: تفاوت در ویژگی‌های سبک بین تصویر تولید شده و تصویر سبک را اندازه گیری می‌کند. فرآیند بهینه سازی:

هدف به حداقل رساندن ضرر کل است که ترکیبی از زیان محتوا و سبک است. یک الگوریتم بهینه سازی (ممدوحاً نزول گرادیان) برای تنظیم مقادیر پیکسل تصویر تولید شده برای به حداقل رساندن این تلفات استفاده می‌شود.

تصویر تولید شده:

خروجی نهایی تصویری است که ساختار محتوا تصویر محتوا را حفظ می‌کند اما سبک هنری تصویر سبک را می‌پذیرد. انتقال سبک کاربردهای مختلفی دارد، از جمله ایجاد تصاویر هنری، تبدیل عکس‌ها به سبک نقاشی‌های معروف و حتی تولید آثار هنری بدیع.

با معرفی مقاله اصلی "A Neural Algorithm of Artistic Style" توسط گاتیس و همکاران، محبوبیت قابل توجهی به دست آورد[11]. در سال 2015. از آن زمان، محققان و توسعه دهندگان تکنیک‌های مختلف برای انتقال سبک را بررسی و بهبود بخشیده اند و آن را به منطقه‌ای جذاب در تقاطع هنر و فناوری تبدیل کرده اند. بسیاری از مسائل کلاسیک را می‌توان به عنوان وظایف تبدیل تصویر قاب کرد، جایی که یک سیستم مقداری تصویر ورودی را دریافت می‌کند و آن را به یک تصویر خروجی تبدیل می‌کند. نمونه‌هایی از پردازش تصویر عبارتند از حذف نویز، وضوح فوق العاده، و رنگ، که در آن ورودی یک تصویر تخریب شده (نویز، با وضوح پایین یا مقیاس خاکستری) و خروجی یک تصویر رنگی با کیفیت بالا است. نمونه‌هایی از بینایی کامپیوتری شامل تقسیم بندی معنایی و تخمین عمق است، که در آن ورودی یک تصویر رنگی است و تصویر خروجی

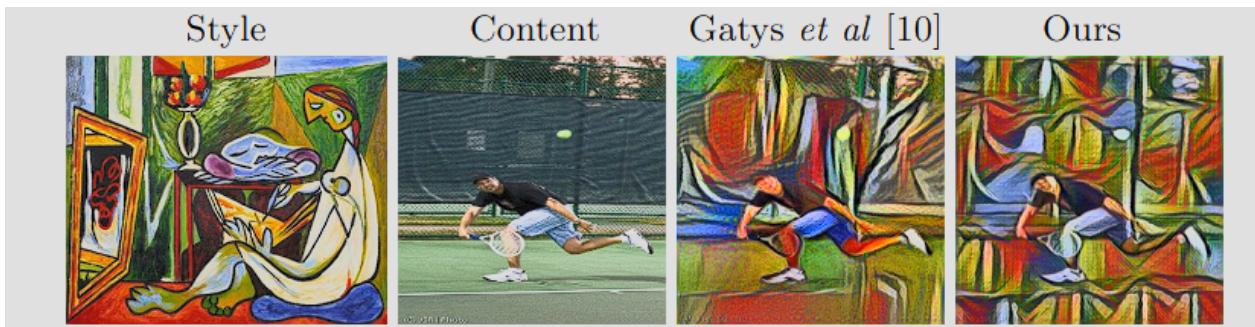
Style transfer¹

Content loss²

Style loss³

اطلاعات معنایی یا هندسی صحنه را رمزگذاری می کند. یک روش برای حل وظایف تبدیل تصویر، آموزش یک شبکه عصبی کانولوشنال پیشخور به شیوه ای ناظارت شده، با استفاده از هر پیکسل است.

تابع ضرر برای لندازه‌گیری تفاوت بین تصاویر خروجی و واقعی. این رویکرد به عنوان مثال توسط دونگ و همکاران برای وضوح فوق العاده استفاده شده است [12]، توسط چنگ و همکاران برای رنگ آمیزی [13]، توسط لانگ و همکاران برای تقسیم بندی [38]، و توسط Eigen و همکاران برای پیش‌بینی نرمال عمق و سطح [40,39]. چنین رویکردهایی در زمان آزمایش کارآمد هستند و فقط نیاز به عبور از شبکه آموزش دیده دارند. با این حال، تلفات هر پیکسل مورد استفاده در این روش‌ها، تفاوت‌های ادراکی بین تصاویر خروجی و واقعی را نشان نمی‌دهد.



شکل(۹-۳): نمونه ای از انتقال سبک [۴۱]

۵-۳- الگوریتم BSP

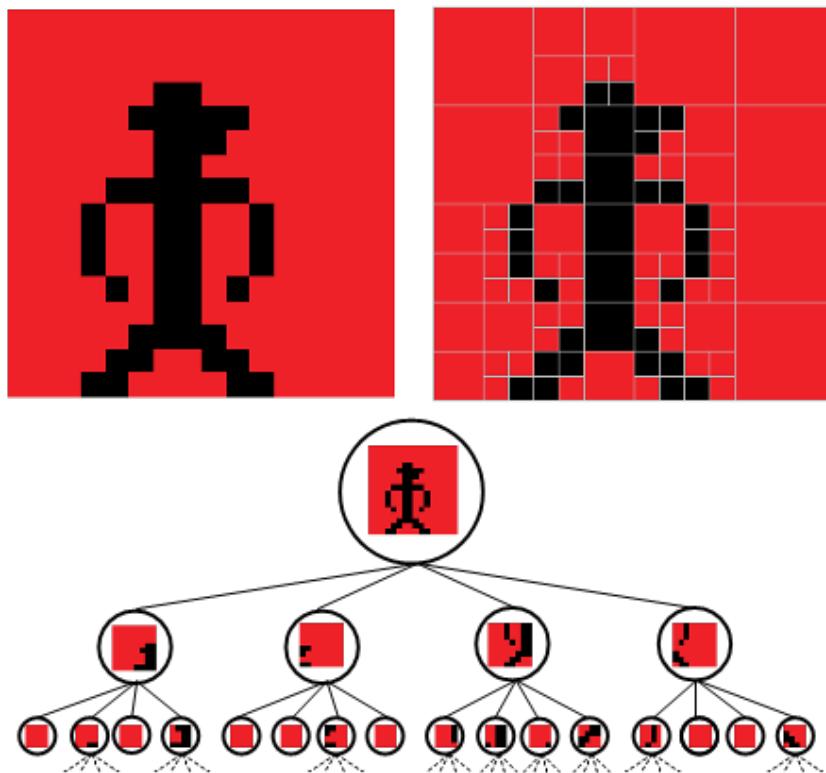
الگوریتم جداساز دودوبی فضا همانند نام خود، یک جداسازی در فضا ایجاد می‌کند، یعنی تقسیم فضای دو بعدی یا سه بعدی به زیرمجموعه‌های مجزا، به طوری که هر نقطه در فضا دقیقاً در یکی از این زیر مجموعه‌ها (که سلول نیز نامیده می‌شوند) قرار می‌گیرد. الگوریتم‌های BSP اغلب به صورت سلسله مراتبی عمل می‌کنند: هر سلول در یک پارتیشن فضایی با اعمال همان الگوریتم به صورت بازگشتی تقسیم‌بندی می‌شود. این عمل باعث می‌شود پارتیشن‌های فضایی به‌اصطلاح در درخت BSP مرتب شوند. علاوه بر این، چنین ساختار داده درختی اجازه می‌دهد تا اطلاعات هندسی هر سلول به سرعت بدست بیاید.

از طریق BSP، فضا را می‌توان به عنوان یک درخت دودوبی به نام درخت BSP نشان داد. چنین الگوریتم‌هایی شامل درخت‌های چهارتایی^۱ و درخت‌های هشتتایی آمی‌باشند: یک درخت چهارتایی فضای دو بعدی را به چهار ربع و یک درخت هشتتایی فضای سه بعدی را به هشت اکтанت تقسیم می‌کند. ما از درخت‌های چهارتایی بر روی تصاویر دو بعدی به عنوان ساده‌ترین مثال استفاده خواهیم کرد، اگرچه همین اصول برای درخت هشتتایی، در فضای سه بعدی و برای انواع دیگر داده‌های ذخیره‌شده اعمال می‌شود.

در حالی که ربع‌های درخت‌های چهارتایی می‌توانند هر شکل مستطیلی داشته باشند، معمولاً مربع‌هایی با اندازه مساوی هستند. یک درخت چهارتایی با عمق n می‌تواند هر تصویر بازیزی $2^n \times 2^n$ پیکسل را نشان دهد، اگرچه تعداد کل گره‌های درخت (و عمق آن) به ساختار تصویر بستگی دارد. گره ریشه کل تصویر را نشان می‌دهد و چهار فرزند آن ربع بالا سمت چپ، بالا سمت راست، پایین سمت چپ و پایین سمت راست تصویر را

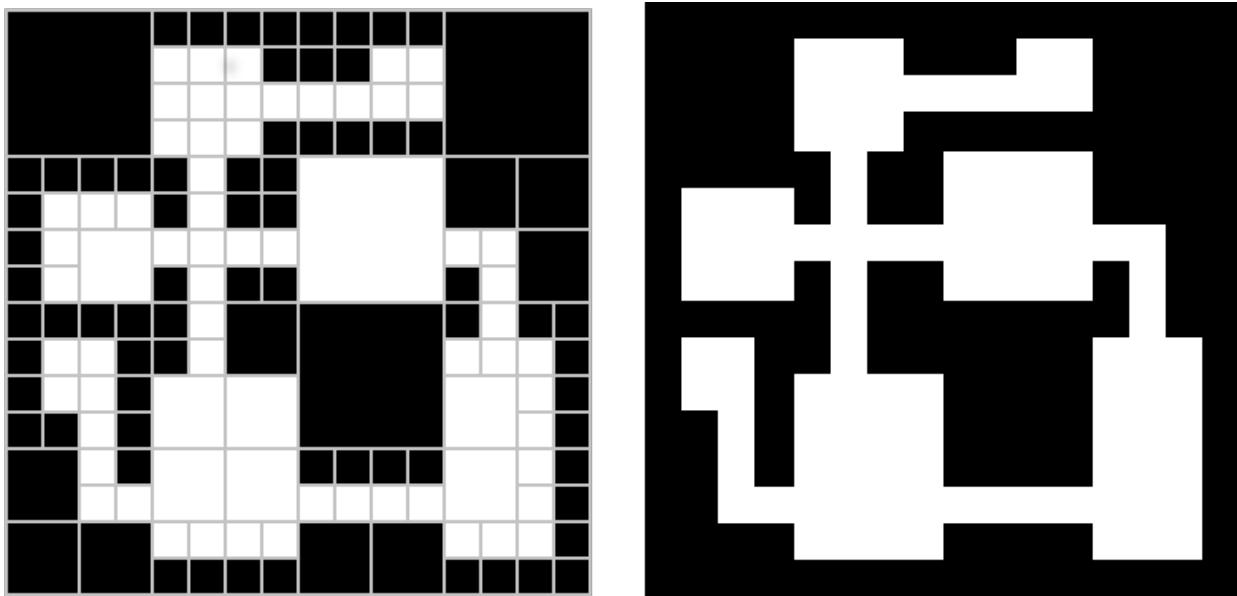
quadtree¹
octree²

نشان می دهند. اگر پیکسل های داخل هر ربع رنگ های متفاوتی داشته باشند، آن ربع تقسیم می شود. این فرآیند به صورت بازگشتی اعمال می شود تا زمانی که هر ربع برگ ۱ (صرف نظر از اندازه) فقط دارای

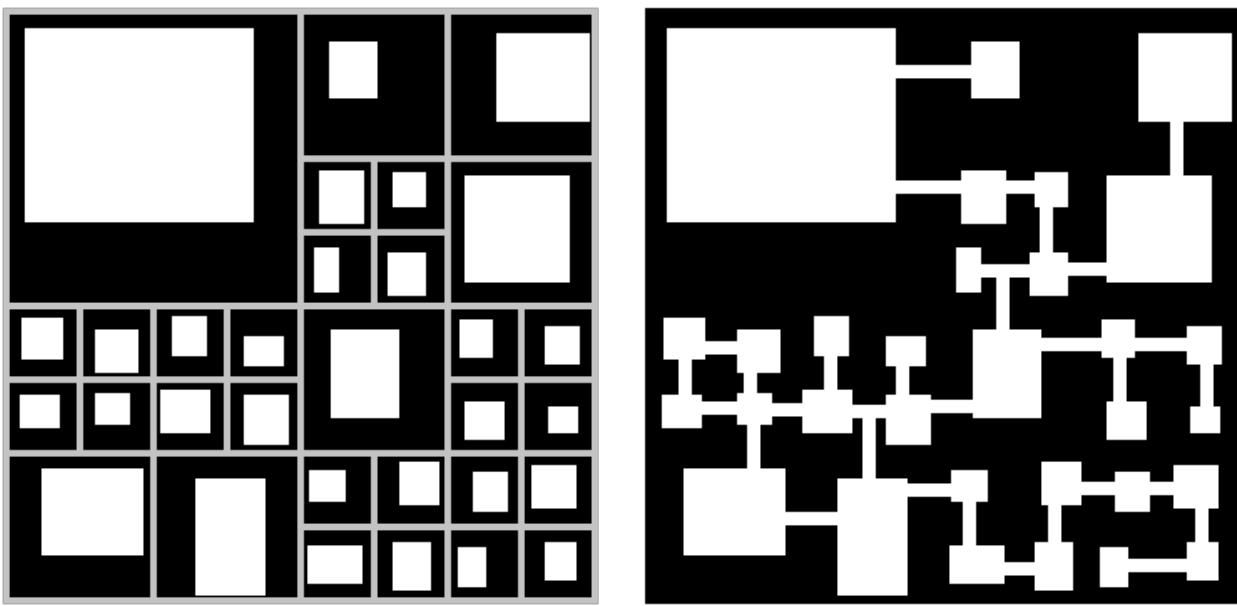


شکل(۳-۱۰): نمونه ای از جداسازی درخت چهارتایی یک تصویر باینری (۰ به صورت قرمز، ۱ به صورت سیاه نشان داده است). مناطق بزرگ شامل یک رنگ، مانند آنهایی که در لبه سمت راست تصویر قرار دارند، بیشتر تقسیم بندی نمی شوند. تصویر ۱۶ در ۱۶ پیکسل است، بنابراین درخت چهارتایی دارای عمق ۴ است. در حالی که یک درخت چهارتایی کاملاً منبسط شده (با گره های برگ حاوی اطلاعات مربوط به یک پیکسل منفرد) دارای ۲۵۶ گره برگ است، مناطق بزرگ یک رنگ منجر به یک درخت چهارتایی با ۹۶ گره برگ می شود. اولین لایه های درخت در پایین نشان داده شده است: گره ریشه شامل کل تصویر است، با چهار فرزند به ترتیب: ربع بالا سمت چپ، ربع بالا سمت راست، ربع پایین سمت چپ، ربع پایین سمت راست (اگرچه ترتیب های دیگر ممکن است).[۲]

الگوریتم های BSP در تصاویر دوبعدی یا سه بعدی استفاده می شوند. این اصل که جداسازی فضا منجر به زیرمجموعه های مجزا و بدون مناطق همپوشانی می شود، به ویژه برای ایجاد اتاق ها در هزارتو یا به طور کلی، مناطق مجزا در سطح بازی مناسب است. کل منطقه هزارتو با گره ریشه درخت BSP نشان داده می شود و به صورت بازگشتی تقسیم می شود تا زمانی که یک شرط پایانی برآورده شود (مانند اندازه حداقل برای اتاق ها). الگوریتم BSP تضمین می کند که هیچ دو اتاق روی هم قرار نخواهند گرفت و ظاهر بسیار ساختار یافته هزارتو را امکان پذیر می کند. اینکه چگونه الگوریتم های مولد از اصول الگوریتم های BSP سنتی پیروی می کند، بر ظاهر هزارتو ایجاد شده تأثیر می گذارد. به عنوان مثال، یک هزارتو را می توان از یک درخت چهارتایی با انتخاب ربع به صورت تصادفی و تقسیم آنها ایجاد کرد. پس از تکمیل، به هر ربع می توان مقدار ۰ (خلال) یا ۱ (اتاق) اختصاص داد، با توجه به اینکه همه اتاق ها به هم متصل هستند. این هزارتو های بسیار متقارن و مربعی مانند آنچه در زیر مشاهده می شود ایجاد می کند.



شکل(۱۱-۳): یک هزارتو که با استفاده از یک درخت چهارتایی ایجاد شده است که هر سلوول کاملاً از فضای خالی (سیاه) یا اتاق (سفید) تشکیل شده است.^[۱۸]

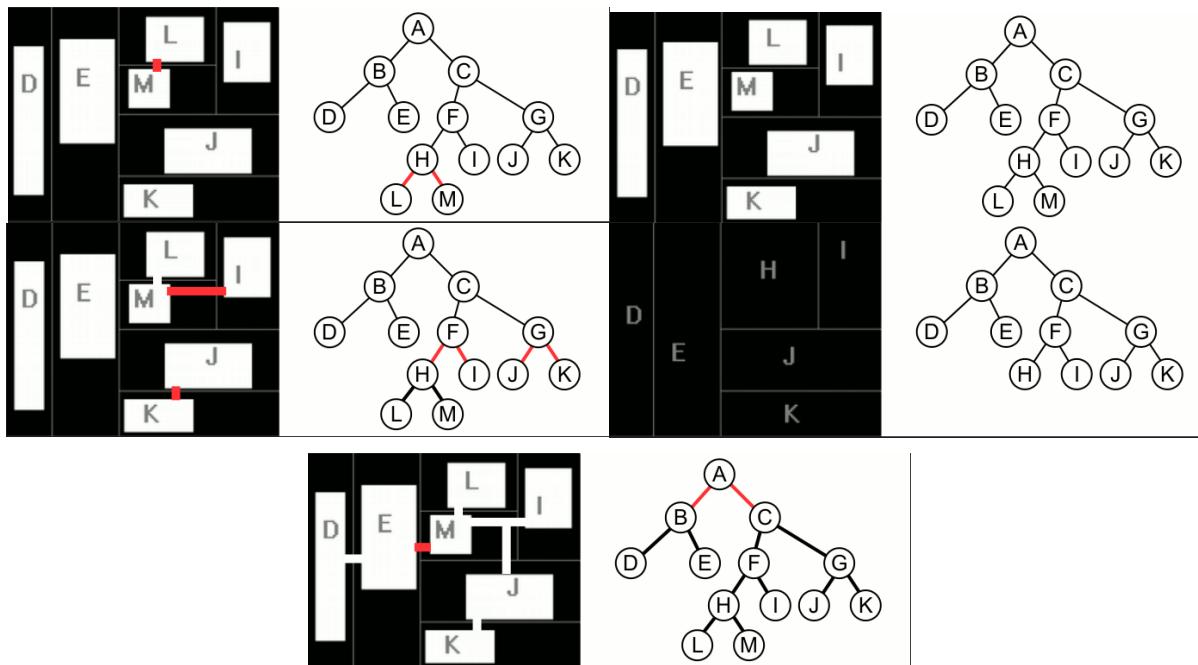


شکل(۱۲-۳): یک هزارتو که با استفاده از یک درخت چهارتایی ایجاد شده است، اما هر ربع حاوی یک اتاق واحد (به صورت تصادفی) و فضای خالی است. پس از تکمیل فرآیند جداسازی، راهروها اضافه می‌شوند.^[۲]

اکنون یک رویکرد تصادفی‌تر را بر اساس تکنیک‌های BSP توصیف می‌کنیم. ابتدا فضایی را برای هزارتو خود، با عرض w و ارتفاع h در نظر می‌گیریم که در گره ریشه درخت BSP ذخیره می‌شود. فضا را می‌توان در امتداد خطوط عمودی یا افقی تقسیم کرد و سلوول‌های جداسده حاصل نیازی به اندازه مساوی ندارند. هنگام تولید درخت، در هر تکرار یک گره برگ به طور تصادفی انتخاب می‌شود و در امتداد یک خط عمودی یا افقی به طور تصادفی انتخاب می‌شود. اگر یک گره برگ کمتر از حداقل اندازه باشد بیشتر از این تقسیم نمی‌شود (برای این مثال حداقل عرض $w=4$ و حداقل ارتفاع $h=4$ را در نظر می‌گیریم). در پایان، هر سلوول پارتيشن شامل یک تک اتاق است. گوشه‌های هر اتاق به طور تصادفی انتخاب می‌شوند تا اتاق در داخل پارتيشن قرار گیرد و اندازه قابل قبولی داشته

باشد (یعنی خیلی کوچک نباشد). هنگامی که درخت تولید می‌شود، راهروهایی با اتصال فرزندان یک والدین به یکدیگر ایجاد می‌کند.
در قسمت زیر یک نمونه از شبه کد این الگوریتم را می‌بینیم:

- 1: start with the entire dungeon area (root node of the BSP tree)
- 2: divide the area along a horizontal or vertical line
- 3: select one of the two new partition cells
- 4: if this cell is bigger than the minimal acceptable size:
- 5: go to step 2 (using this cell as the area to be divided)
- 6: select the other partition cell, and go to step 4
- 7: for every partition cell:
- 8: create a room within the cell by randomly choosing two points (top left and bottom right) within its boundaries
- 9: starting from the lowest layers, draw corridors to connect rooms in the nodes of the BSP tree with children of the same parent
- 10: repeat 9 until the children of the root node are connected



شکل(۱۳-۳): تصویرسازی جداسازی تصادفی یک منطقه هزارتو A، که در گره ریشه درخت BSP موجود است: A از طریق یک خط عمودی به B و C تقسیم می‌شود (مختصات X آن به طور تصادفی تعیین می‌شود). تا زمانی که برگها بالاتر از یک اندازه خاص باشند به طور مکرر بیشتر شکافته می‌شوند. پس از ایجاد درخت، اتاق‌ها و راهروها قرار می‌گیرند: برای هر گره برگ در درخت BSP، یک اتاق با انتخاب تصادفی مختصات گوشه‌های سمت راست بالا و پایین، در محدوده سلول پارتبیشن قرار می‌گیرد. راهرو گره‌هایی را به هم متصل می‌کند که یک والد مشترک دارند.[۲]

در حالی که BSP در اینجا عمدهاً برای ایجاد اتاق‌های غیرهمپوشان استفاده می‌شد، باید توجه داشت که سلسله مراتب درخت BSP را می‌توان برای سایر جنبه‌های تولید هزارتو نیز استفاده کرد. تصویر بالا نشان می‌دهد که چگونه اتصال اتاق را می‌توان توسط درخت BSP تعیین کرد: استفاده از راهروها برای اتصال اتاق‌های متعلق به یک والدین، شناس همپوشانی یا متقاطع راهروها را کاهش می‌دهد.

علاوه بر این، سلوولهایی که والد هستند را می‌توان برای تعریف گروههایی از اتاق‌ها با موضوع یکسان استفاده کرد. به عنوان مثال، بخشی از هزارتو ممکن است حاوی هیولاهاي سطح بالاتر یا هیولاهاي باشد که در برابر ضربه آسیب‌پذیرتر هستند. همچنین با اتصال راهرو بر اساس سلسله مراتب درخت BSP، این گروه از اتاق‌ها ممکن است یک ورودی واحد از بقیه هزارتو داشته باشند. این می‌تواند این امکان را بدهد تا چنین اتاقی به عنوان یک زندان یا به عنوان یک منطقه با نور کمتر تزئین شود.



شکل(۱۴-۳): نمونه هزارتوی تصویر بالا، با استفاده از جداسازی‌ها برای موضوع‌بندی محتوای اتاق. سلوولهای پارتیشن B و C فقط توسط یک راهرو به هم متصل می‌شوند. این اجازه می‌دهد تا اتاق‌های پارتیشن B قفل شوند (قفل سبز)، که برای دسترسی به آن نیاز به کلید از سلوول C دارد (اتاق L). به طور مشابه، اتاق‌های سلوول B فقط حاوی گنجها و پاداش‌ها هستند، در حالی که اتاق‌های پارتیشن C عمدها حاوی هیولا هستند. علاوه بر این، رتبه چالش هیولاها در سلوول C بین گرههای فرزند آن تقسیم می‌شود: پارتیشن G حاوی هیولاهای ضعیف است در حالی که سلوول F حاوی هیولاهای چالش‌برانگیز با قدرت‌های جادویی است.^[۲]

۶-۳- نتیجه‌گیری

در این فصل به بررسی کارهای انجام شده در حوزه تولید هزارتوها و پردازش و تغییر سبک تصویر پرداختیم. با مطالعه آن‌ها این نتیجه کسب شد که در اکثر موارد خروجی الگوریتم‌ها شباهت کمی به یکدیگر دارند و هر الگوریتم برای استفاده در بازی‌های متفاوتی مناسب است. در فصل بعد با کمک گرفتن از دو حوزه پردازش تصویر و تولید مرحله روشی ترکیبی را ارائه خواهیم کرد.

فصل 4

روش پیشنهادی

۱-۴- مقدمه

در این پروژه قصد داریم با استفاده از پردازش تصویر برنامه‌ای بسازیم که بتواند با استفاده از تصویر نقشه داده شده به برنامه فایل سه بعدی یا دو بعدی نقشه‌ی اولیه را به ما بدهد این برنامه با ساده کردن و کوتاه کردن زمان این فرایند (ساخت نقشه اولیه) به طراحان بازی این قابلیت را میدهد که از زمان خود به صورت بهینه تری استفاده کنند.

۲-۴- مدل کلی روش ارائه شده

عملکرد روش ارائه شده به این شکل است که ابتدا با استفاده از رابط کاربری پیاده سازی شده کاربر میتواند نقشه‌ی اسکن شده‌ی خود را به برنامه بدهد و سپس با مشخص کردن و وارد کردن المان‌های در نظر داشته شده به لیست به عنوان قالب و الگو و همچنین وارد کردن صورت نهایی آن الگوهای در رابط برنامه پردازش را به روی نقشه‌ی اسکن شده انجام میدهد و پس از تشخیص الگوهای در عکس فایل دو بعدی یا سه بعدی خواسته شده را میسازد و سپس صورت نهایی الگوهای را در موقعیت خواسته شده قرار میدهد و فایل را به عنوان خروجی در محل خواسته شده ذخیره میکند.

۳-۴- پیاده سازی سرکوب غیر حداقلی

این کد الگوریتم سرکوب غیر حداقلی پیشنهاد شده توسط Malisiewicz و همکاران را پیاده سازی می‌کند. برای حذف جعبه‌های محدود کننده اضافی که به طور قابل توجهی همپوشانی دارند. این عملکرد به طور موثر جعبه‌های محدود کننده اضافی را سرکوب می‌کند، و تضمین می‌کند که فقط مرتبطترین آنها حفظ می‌شوند، که در کارهایی مانند تشخیص شی برای جلوگیری از تشخیص‌های تکراری یک شی ضروری است.

تعريف تابع:

کد تابعی به نام non_max_suppression_fast را تعریف می‌کند که دو پارامتر را می‌گیرد: آرایه‌ای از جعبه‌های محدود کننده^۱ و مقدار آستانه برای تعیین مقدار همپوشانی مورد نیاز برای سرکوب.^۲

```
def non_max_suppression_fast(boxes, overlapThresh):
```

جابجایی جعبه‌های خالی:

اگر هیچ کادر محدود کننده‌ای وجود نداشته باشد ($\text{len}(\text{boxes}) == 0$), تابع یک لیست خالی برمی‌گرداند.

```
if len(boxes) == 0:
    return []
```

Boxes^۱

overlapThresh^۲

تبدیل انواع داده های جعبه:

اگر مختصات جعبه مرزی اعداد صحیح باشند ("i" dtype)، تابع آنها را به شناور تبدیل می کند. این تبدیل برای عملیات تقسیم دقیق ضروری است.

```
if boxes.dtype.kind == "i":  
    boxes = boxes.astype("float")
```

مقداردهی اولیه:

یک لیست خالی به نام pick را برای ذخیره شاخص های جعبه های مرزی انتخاب شده مقداردهی اولیه می کند.

استخراج مختصات جعبه:

مختصات کادرهای مرزی (X بالا-چپ، y بالا-چپ، X پایین-راست، پایین-راست y) از آرایه جعبه ها استخراج می شود.

```
x1 = (parameter  
y1 =  
x2 = boxes[:,2]  
y2 = boxes[:,3]
```

محاسبه نواحی جعبه مرزی:

مساحت جعبه های مرزی با استفاده از فرمول محاسبه می شود: مساحت = $(x2 - x1 + 1) * (y2 - y1 + 1)$

```
area = (x2 - x1 + 1) * (y2 - y1 + 1)
```

مرتب سازی جعبه های مرزبندی:

جعبه های مرزی بر اساس مختصات y پایین سمت راست با استفاده از np.argsort مرتب می شوند.

```
idxs = np.argsort(y2)
```

فرآیند تکراری:

تابع بر روی شاخص های مرتب شده جعبه های مرزی تکرار می شود. در هر تکرار، آخرین شاخص (i) را در لیست مرتب شده انتخاب می کند و آن را به فهرست شاخص های انتخاب شده اضافه می کند.

مختصات (xx1, yy1, xx2, yy2) تقاطع بین کادر محدود انتخابی و تمام جعبه های مرزبندی قبلی را محاسبه می کند. عرض (W) و ارتفاع (H) ناحیه تقاطع را محاسبه می کند. نسبت همپوشانی بین منطقه تقاطع و مساحت جعبه های مرزی قبلى را محاسبه می کند. ایندکس هایی را از فهرست شاخص هایی که همپوشانی قابل توجهی با کادر محدود انتخاب شده بر اساس همپوشانی Thresh دارند حذف می کند. این روند تا زمانی ادامه می یابد که هیچ شاخصی در لیست باقی نماند. در نهایت، تابع زیرمجموعه کادرهای محدودی را که انتخاب شده و به اعداد صحیح تبدیل شده اند، بر می گرداند.

```

while len(idxs) > 0:

    last = len(idxs) - 1
    i = idxs[last]
    pick.append(i)

    xx1 = np.maximum(x1[i], x1[idxs[:last]])
    yy1 = np.maximum(y1[i], y1[idxs[:last]])
    xx2 = np.minimum(x2[i], x2[idxs[:last]])
    yy2 = np.minimum(y2[i], y2[idxs[:last]])

    w = np.maximum(0, xx2 - xx1 + 1)
    h = np.maximum(0, yy2 - yy1 + 1)

    overlap = (w * h) / area[idxs[:last]]
    idxs = np.delete(idxs, np.concatenate(([last],
                                           np.where(overlap > overlapThresh)[0])))
```

۴-۴-۴- پیاده‌سازی پردازش تصویر

ما با استفاده از کتابخانه OpenCV تطبیق الگو را بین یک تصویر ورودی و یک تصویر الگو انجام می‌دهیم. اجازه دهید اجزای اصلی و مراحل کد را بررسی کنیم:

۴-۱-۴- کتابخانه های استفاده شده

- cv2 : وارد کردن کتابخانه OpenCV برای پردازش تصویر.
- numpy : وارد کردن NumPy برای عملیات عددی.
- imutil : وارد کردن imutil برای عملکردهای راحت پردازش تصویر.
- nms : وارد کردن یک تابع سرکوب غیر حداکثری از یک مازول سفارشی به نام nms. این تابع برای جلوگیری از همپوشانی جعبه های مرزی استفاده می شود.

۴-۲- تابع پیش پردازش

```

5
6     def preprocess_image(image):
7         gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
8         blurred = cv2.GaussianBlur(gray, (5, 5), 0)
9         edges = cv2.Canny(blurred, 50, 200)
10
11     return edges
```

عملکرد با تبدیل تصویر ورودی به مقیاس خاکستری شروع می شود. تصاویر در مقیاس خاکستری فقط حاوی اطلاعات شدت هستند، که پردازش را ساده می کند و پیچیدگی محاسباتی را کاهش می دهد. این مرحله با

حفظ ویژگی های مهم تصویر، اطلاعات رنگ را حذف می کند. تاری گاووسی روی تصویر خاکستری اعمال می شود. این یک تکنیک رایج برای کاهش نویز و صاف کردن جزئیات در تصویر است. پارامترهای (5 و 5) اندازه هسته مورد استفاده برای محو کردن را نشان می دهد و 0 نشان دهنده انحراف استاندارد توزیع گاووسی است که میزان تاری را تعیین می کند. تشخیص لبه کنی¹ یک تکنیک محبوب برای تشخیص لبه ها در یک تصویر است. با شناسایی مناطقی با گرادیان شدت قابل توجه، که اغلب با مرزهای شی یا ویژگی ها مطابقت دارد، کار می کند. پارامترهای 50 و 200 به ترتیب آستانه های پایین و بالا را برای تشخیص لبه نشان می دهند. این مقادیر حساسیت آشکارساز لبه را تعیین می کند. در نهایت،تابع تصویر حاصل از لبه را برمی گرداند. این تصویر فقط حاوی لبه های استخراج شده از تصویر اصلی در مقیاس خاکستری است که برای عملیات تطبیق الگوی بعدی ضروری است.

۴-۳-۴- تابع تطبیق

تابع تطبیق، تطبیق الگو را در مقیاس های چندگانه انجام می دهد، تطابق های بالقوه را تعیین می کند، برای اصلاح نتایج، از مهار غیرحداکثری استفاده می کند، و کادرهای مرزی را در اطراف مطابقت های شناسایی شده روی تصویر منبع ترسیم می کند. این به طور موثر نمونه هایی از الگو را در تصویر منبع شناسایی می کند، و وظایف تشخیص شی و محلی سازی را تسهیل می کند.

پیش پردازش تصاویر الگو و منبع:

این تابع با پیش پردازش هر دو تصویر الگو و منبع با استفاده از تابع preprocess_image() شروع می شود. این مرحله پیش پردازش تضمین می کند که هر دو تصویر به طور مناسب برای تطبیق الگو آماده شده اند. پیش پردازش شامل تبدیل تصاویر به مقیاس خاکستری، اعمال تاری گاووسی برای کاهش نویز و انجام تشخیص لبه با استفاده از الگوریتم کنی است.

```
template_processed = preprocess_image(template)
image_processed = preprocess_image(image)
```

تطبیق قالب چند مقیاسی:

سپس تابع به انجام تطبیق قالب چند مقیاسی ادامه می دهد. در طیف وسیعی از مقیاس ها تکرار می شود و اندازه تصویر منبع را متناسب با آن تغییر می دهد. هدف از تطبیق چند مقیاسی تشخیص اشیاء در اندازه های مختلف در تصویر منبع است. این مهم است زیرا اشیا ممکن است در مقیاس های مختلف به دلیل تغییر در فاصله، پرسپکتیو یا وضوح تصویر ظاهر شوند.

```

for scale in np.linspace(0.5, 1.5, 25)[::-1]:
    resized = imutils.resize(image_processed, width=int(image_processed.shape[1] * scale))
    r = image_processed.shape[1] / float(resized.shape[1])

```

تطبیق الگو:

در هر مقیاس، تابع تطبیق الگو را با استفاده از تابع `cv2.matchTemplate()` اعمال می کند. این تابع تصویر الگو را با بخشی از تصویر منبع در مقیاس فعلی مقایسه می کند و شباهت بین آنها را محاسبه می کند. نتیجه یک نقشه شباهت است که نشان می دهد الگو چقدر با مناطق مختلف تصویر منبع مطابقت دارد.

```
result = cv2.matchTemplate(resized, template_processed, cv2.TM_CCOEFF_NORMED)
```

آستانه و مکان یابی مسابقات:

نقشه شباهت برای شناسایی تطابقات بالقوه بین الگو و تصویر منبع آستانه گذاری شده است. در این کد از آستانه 0.8 استفاده شده است، به این معنی که تنها مناطقی که دارای امتیاز شباهت بالاتر از این آستانه هستند به عنوان منطبق در نظر گرفته می شوند. سپس تابع `np.where` برای تعیین موقعیت این تطابق ها در نقشه شباهت استفاده می شود.

```
loc = np.where(result >= 0.8)
```

نسل جعبه مرزی:

برای هر تطابق یافت شده، تابع مختصات جعبه مرزی مربوطه را در تصویر منبع اصلی محاسبه می کند. این کار را با مقیاس گذاری مختصات مطابقت بر اساس ضریب مقیاس فعلی و ابعاد تصویر الگو انجام می دهد. این کادرهای محدود کننده مناطقی را در تصویر مبدأ نشان می دهند که در آن الگو شناسایی شده است.

```
matches.append((int(pt[0] * r), int(pt[1] * r), int((pt[0] + tW * r), int((pt[1] + tH * r))))
```

سرکوب غیر حداقلی (NMS):

برای حذف جعبه های کران کننده اضافی یا همپوشانی، این تابع با استفاده از تابع `nms` وارد شده از مژول `non_max_suppression_fast` سرکوب غیر حداقلی (NMS) را اعمال می کند. NMS مطمئن ترین جعبه های مرزی را انتخاب می کند در حالی که تشخیص های ضعیفتر و همپوشانی را سرکوب می کند. این به پالایش مجموعه نهایی مطابقت های شناسایی شده کمک می کند.

```
pick = non_max_suppression_fast(np.array(matches), 0.3)
```

ترسیم جعبه های مرزبندی:

در نهایت، این تابع کادرهای محدود کننده را در اطراف مطابقت های شناسایی شده روی تصویر منبع اصلی ترسیم می کند. این کادرهای محدود کننده به صورت بصری مناطقی را که الگو در آن یافت می شود برجسته می کنند.

```
for (startX, startY, endX, endY) in pick:
    cv2.rectangle(image, (startX, startY), (endX, endY), (255, 0, 0), 3)
```

۴-۵-پیاده سازی ساخت نقشه سه بعدی

این اسکریپت فرآیند وارد کردن اشیا از چندین فایل بلندر، سازماندهی آنها در مجموعه ها، قرار دادن آنها در صحنه و ذخیره صحنه نهایی به عنوان یک فایل بلندر جدید را خودکار می کند. این یک ابزار مناسب برای تولید صحنه های پیچیده مت Shank از اشیاء از منابع مختلف است.

پاک کردن اشیاء مش موجود:

قبل از وارد کردن اشیاء جدید، اسکریپت همه اشیاء مش موجود در صحنه را از حالت انتخاب خارج کرده و حذف می کند تا از یک صفحه تمیز مطمئن شود.

```
bpy.ops.object.select_all(action='DESELECT')
bpy.ops.object.select_by_type(type='MESH')
bpy.ops.object.delete()
```

ایجاد یک فایل بلندر جدید:

صحنه فعلی را به عنوان یک فایل Blender جدید ذخیره می کند، که به عنوان نقطه شروع برای وارد کردن اشیا عمل می کند.

```
bpy.ops.wm.save_as_mainfile(filepath=output_path)
```

وارد کردن اشیا از فایل های ترکیبی:

برای هر هدف مشخص شده در targets_dict، اسکریپت اشیاء را از فایل های Blender مربوطه وارد می کند. کل بلوک داده را از هر فایل ترکیبی بارگیری می کند و اشیاء را به صحنه فعلی اضافه می کند. یک مجموعه جدید برای هر فایل ترکیبی ایجاد می شود و اشیاء وارد شده به این مجموعه ها پیوند داده می شوند. موقعیت اشیا با توجه به موقعیت های مشخص شده در targets_dict تنظیم می شود.

```
for target in targets_dict:

    blend_file = target["path"]
    position = target["pos"]

    with bpy.data.libraries.load(blend_file) as (data_from, data_to):
        data_to.objects = data_from.objects

    new_collection = bpy.data.collections.new(
        name=os.path.basename(blend_file))
    bpy.context.scene.collection.children.link(new_collection)

    for obj in data_to.objects:
        if obj is not None:
            new_collection.objects.link(obj)

    for obj in new_collection.objects:
        obj.location = position
```

ذخیره فایل ترکیب نهایی:

هنگامی که همه اشیاء وارد و قرار گرفتند، اسکریپت صحنه نهایی را به عنوان یک فایل Blender در مسیر خروجی مشخص شده ذخیره می کند.

```
bpy.ops.wm.save_as_mainfile(filepath=output_path)
```

۶-۴- پیاده سازی ساخت نقشه دو بعدی

اسکریپت چندین تصویر را بر روی یک تصویر پایه قرار می دهد، ترکیب حاصل را ذخیره می کند و بازخورد را به کاربر ارائه می دهد. این دستکاری و ترکیب اولیه تصویر را با استفاده از پایتون و OpenCV نشان می دهد.

تابع اصلی :

این تابع نقطه ورود اصلی اسکریپت است. فهرستی از دیکشنری های حاوی مسیرها و موقعیت های تصویر را تعریف می کند. سپس تابع `insert_images()` را با این لیست به عنوان آرگومان فراخوانی می کند. پس از درج تصاویر بر روی تصویر پایه، نتیجه را به عنوان "results/map.png" ذخیره می کند و یک پیام موفقیت آمیز چاپ می کند.

تابع وارد کردن عکس ها:

این تابع فهرستی از دیکشنری ها و اندازه خروجی را به عنوان ورودی می گیرد. این یک تصویر پایه پر از رنگ سفید بر اساس اندازه خروجی مشخص شده ایجاد می کند.

تکرار از طریق هر فرهنگ لغت در لیست ورودی:

فایل تصویری را که با کلید "image_path" در فرهنگ لغت مشخص شده است می خواند. اندازه تصویر را به اندازه ثابت 100x100 پیکسل تغییر می دهد. موقعیت را از فرهنگ لغت بازیابی می کند. تصویر تغییر اندازه را روی تصویر پایه در موقعیت مشخص شده قرار می دهد. در نهایت، تصویر پایه به دست آمده را با تصاویر درج شده بر می گرداند.

```
def insert_images(image_dict_list, output_size=(800, 600)):
    base_image = 255 * np.ones((output_size[1], output_size[0], 3), dtype=np.uint8)
    for image_info in image_dict_list:
        image = cv2.imread(image_info["image_path"])
        position = image_info["position"]
        image = cv2.resize(image, (100, 100))
        x, y = position
        base_image[y:y+image.shape[0], x:x+image.shape[1]] = image

    return base_image
```

۷-۴- رابط کاربری

به طور کلی، برنامه یک رابط کاربر پسند برای انجام وظایف تشخیص شکل فراهم می کند و به کاربران امکان می دهد تصاویر منبع را مشخص کنند، الگوها را اضافه کنند، پوشه های خروجی را انتخاب کنند و الگوریتم تشخیص Tkinter شکل را به راحتی اجرا کنند. کلاس تشخیص شکل پنجره برنامه رابط کاربری گرافیکی را با استفاده از مقداردهی اولیه می کند. عنوان پنجره را روی "تشخیص شکل" تنظیم می کند.

ویجت های مختلفی در پنجره برنامه ایجاد و چیده می شوند:
 برچسب های متñی را برای «تصویر منبع»، «تصویر خروجی» و «الگوهای کاربر» نمایش میدهیم. فیلدهای ورودی به کاربران اجازه می دهد متن را وارد یا نمایش دهند (مسیرهای فایل یا مسیرهای پوشه). دکمه ها کاربران را قادر می سازد تا اقداماتی مانند باز کردن یک فایل، انتخاب یک پوشه، افزودن یا حذف الگوها، اجرای تشخیص شکل و لغو را فعال کنند. یک ویجت بوم برای حاوی قاب های قالب استفاده می شود. یک نوار پیمایش به بوم اضافه می شود تا در صورت وجود الگوهای زیاد، پیمایش را فعال کند و فریم های جداگانه برای نگهداری ویجت های مرتبط با هر الگو ایجاد می شوند.

مدیریت الگو:

این برنامه به کاربران اجازه می دهد تا قالب ها را به صورت پویا اضافه یا حذف کنند. هر الگو با یک قاب نشان داده می شود که حاوی ویجت هایی برای انتخاب تصاویر و تعیین گزینه ها است. هنگامی که یک الگو اضافه می شود، یک قاب با گزینه هایی برای مرور یک تصویر یا انتخاب یک تصویر از پیش ذخیره شده ایجاد می شود. مسیرهای تصویر به صورت چند تایی مرتبط با هر فریم الگو ذخیره می شوند.

تعامل کاربر:

کاربران می توانند با رابط کاربری گرافیکی تعامل داشته باشند تا اقدامات مختلفی را انجام دهند مانند باز کردن یک گفتگوی فایل برای انتخاب یک تصویر منبع ، انتخاب یک گفتگوی پوشه برای انتخاب پوشه خروجی ، افزودن یا حذف الگوها، که شامل انتخاب تصاویر یا مشخص کردن گزینه هایی برای هر الگو می شود. اجرای الگوریتم تشخیص شکل، که پردازش تصویر منبع انتخاب شده و الگوها را برای شناسایی اشکال آغاز می کند.

تشخیص شکل:

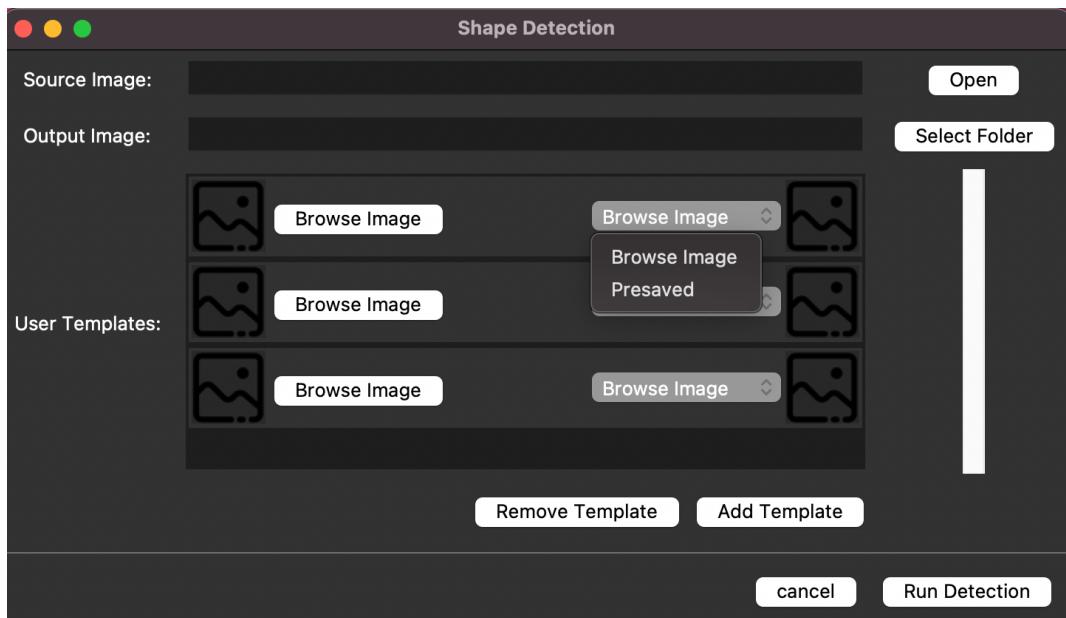
متدها تشخیص زمانی فراخوانی می شود که کاربر روی دکمه "اجرا تشخیص" کلیک کند. اطلاعات لازم مانند مسیر تصویر منبع و مسیر پوشه خروجی را جمع آوری می کند، سپس تابع پیدا کردن شکل ها را از ماژول opencv برای انجام تشخیص شکل فراخوانی می کند. اشکال شناسایی شده در پوشه خروجی مشخص شده ذخیره می شوند.

تغییر اندازه تصویر:

اندازه تصاویر انتخاب شده توسط کاربر قبل از نمایش در برنامه به اندازه ثابت (5050x5050 پیکسل) تغییر می کند. این تضمین می کند که تصاویر به طور یکنواخت در فریم های قالب قرار می گیرند.

اطلاعات رفع اشکال:

این برنامه اطلاعات مربوط به اشکال زدایی را در کنسول چاپ می کند و جزئیات مربوط به فریم های قالب و تاپل های تصویر را ارائه می دهد. این اطلاعات به درک رفتار برنامه و عیب یابی هر گونه مشکل کمک می کند.

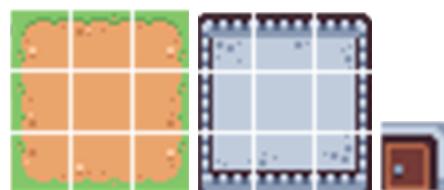


شکل(۱-۴): تصویری از رابط کاربری پیاده سازی شده.

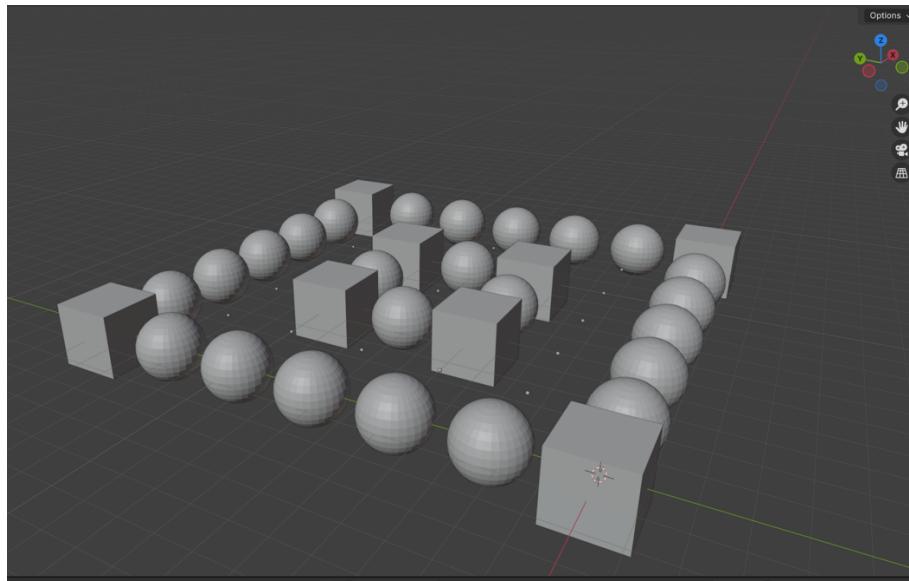
۸-۴- نمونه خروجی



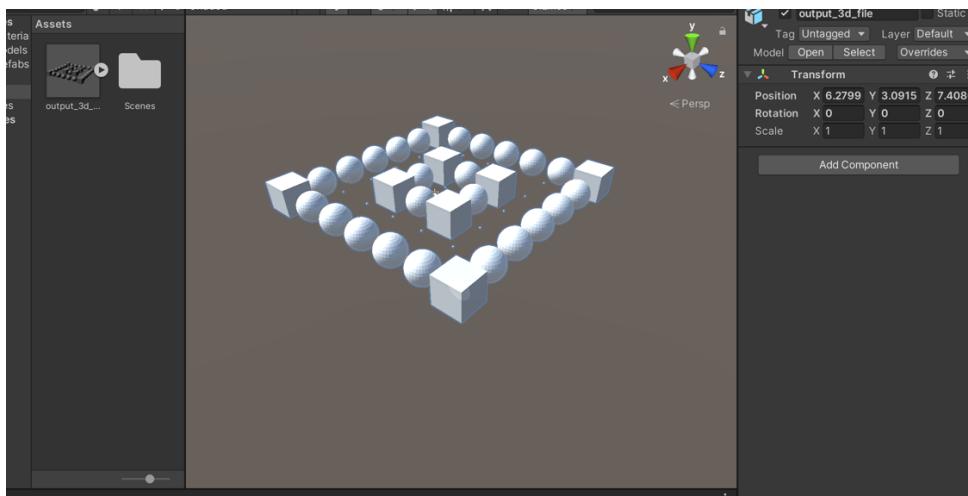
شکل(۲-۴): نمونه خروجی نقشه دو بعدی.



شکل(۳-۴): عکس های داده شده به عنوان قالب های نقشه دو بعدی.



شکل (۴-۴): نمونه خروجی نقشه‌ی سه بعدی.



شکل (۴-۵): نمونه خروجی نقشه‌ی سه بعدی در محیط یونیتی.

۹-۴- نتیجه‌گیری

در این فصل به نحوه‌ی پیاده سازی اپلیکیشن خود برای ساخت نقشه‌ی اولیه پرداختیم. کاربران با استفاده از رابط کاربری میتوانند عکس اسکن شده‌ی خود را از نقشه‌ی اولیه خود (به صورت سه بعدی از بالا به پایین و یا دو بعدی) همراه با قالب‌های خود آپلود کنند و سپس با استفاده از اجرای تشخیص فایل خروجی متناسب با خواسته خود را در محل نظر تایین شده دریافت کنند.

فصل 5:

ارزیابی روش پیشنهادی

۱-۵- مقدمه

در این فصل قصد داریم عملکرد روش ارائه شده در فصل پیشین را با کمک نتایج به دست آمده از پیاده‌سازی انجام شده بررسی کنیم و آن را با روش‌های سنتی دیگر مقایسه و نتایج را ارائه دهیم و راه حل مطرح شده را مورد ارزیابی قرار دهیم.

۲-۵- محیط آزمایش روش ارائه شده

برای ارزیابی و پیاده‌سازی روش ارائه شده در این گزارش از زبان پایتون تحت سیستم‌عامل unix استفاده شده است. سخت افزار استفاده شده بدین شرح است:

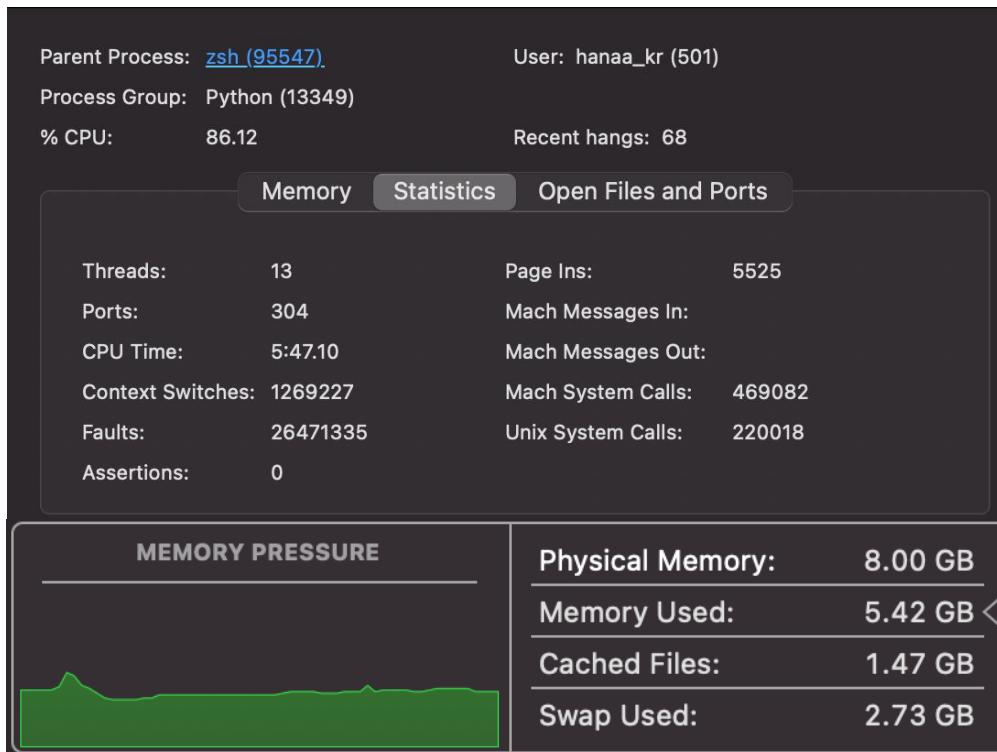
پردازنده Quad-Core Intel Core i5

8 گیگابایت رم

250.69 GB

مچنین برای شبیه‌سازی محیط گرافیکی از محیط بازی‌سازی unity 3D ورژن 2020.3.26 و ابزار blender استفاده شده است.

برای به دست آوردن آمار قابل قیاس از ابزار activity monitor استفاده شده است که میزان دقیق زمان اجرا و حافظه مصرفی را در اختیار ما قرار می‌دهد.



شکل(۱-۵): تصویری از activity monitor

۳-۵- ارزیابی

برای بررسی عملکرد و میزان کارای روش ارائه شده لازم است تا فاکتوری برای ارزیابی آن داشته باشیم که این فاکتور هم در روش پیشنهادی و هم در روش های سنتی دیگری که قرار است با عملکرد روش ما مقایسه شود استفاده می شود، هر چند نمی توان معیار دقیقی برای مقایسه این روش تعیین کرد، چرا برنامه ای مشابه روش ارائه شدهی ما در دسترس نمی باشد و ارزیابی دقیق هنگامی انجام پذیر است که بتوان دو برنامه را از نظر معیار های سیستم با هم مقایسه کرد. حال ما در این قسمت تلاش کردیم با نهایت دقت معیار های قابل اندازه گیری را با هم مقایسه کنیم.

فاکتور و معیارها :

برای مقایسه درست بین روش ها ما نیازمند فاکتوری ثابت داریم تا محیط آزمایش یکسانی را فراهم کنیم. به این منظور ما فاکتوری را نقشه ای تعریف میکنیم که بتوانیم با استفاده از آن ۲ معیار خود را مقایسه کنیم. معیار هایی که مناسب برای مقایسه هستند شامل:

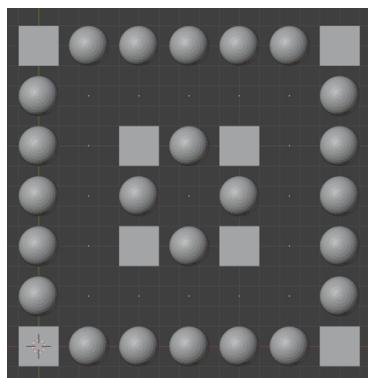
منابع انسانی : تعداد افرادی که برای ساخت و رسیدن به نقشه مورد نظر نیاز داریم.

زمان : مقدار زمانی که (به دقیقه) صرف میشود تا نقشه مورد نظر ما ساخته شود.

حافظه : مقدار حافظه ای اشغال شده هنگام ساخت نقشه.

۴-۵- نتایج مقایسه

پس از معادل سازی و اجرای هر دو روش اعداد به دست آمده، در نهایت به جداول زیر می رسیم که در آن دو روش ساخت نقشه مقایسه شدند، ستون اول نشان دهندهی روش پیشنهادی ما و ستون دوم نشان دهندهی ساخت همان نقشه توسط یک طراح بازی است مقدار زمان مصرف شده با دقیقه و حافظه مصرفی با مگابایت است.



شکل(۳-۵): تصویری از نقشه ای آزمایش شده

ازمایش پیچیدگی:

معیار پیچیدگی نقشه	MapGenerator	MadeByDesigner
time	6.5	23.67
memory	43.46	55.36
منابع انسانی	1	2

جدول (3-5): بالا خروجی ساخت نقشه توسط طراح بازی و ابزار ما

۵-۵- مقایسه ابزار opencv با ابزارهای دیگر

OpenCV یک کتابخانه محبوب برای کارهای بینایی کامپیوترا از جمله تشخیص اشیا است، اما خود الگوریتم تشخیص اشیا خاصی نیست. در عوض، ابزارها و توابعی را برای پیاده سازی الگوریتم‌های مختلف تشخیص اشیا فراهم می‌کند. در اینجا مقایسه OpenCV با برخی از الگوریتم‌های رایج تشخیص اشیا آورده شده است:

آبشرار هار:

OpenCV شامل آبشرارهای Haar است که نوعی الگوریتم تشخیص اشیا بر اساس یادگیری ماشینی است. آبشرار هار از یک سری ویژگی‌های ساده برای تشخیص اشیاء در تصاویر استفاده می‌کند. در مقایسه با سایر الگوریتم‌ها، آبشرارهای هار از نظر محاسباتی کارآمد هستند، اما ممکن است برای کارهای پیچیده تشخیص اشیا دقیق یا همه‌کاره نباشند.

هیستوگرام گرادیان‌های جهت دار^۱ : هیستوگرام گرادیان‌های جهت دار یکی دیگر از الگوریتم‌های محبوب برای تشخیص اشیا، به ویژه برای تشخیص عابران پیاده و پیکره‌های انسان است که بر توزیع جهت گیری‌های گرادیان در یک تصویر مرکز دارد و در تشخیص اشکال و ساختار اشیا موثر است. OpenCV شامل توابعی برای محاسبه ویژگی‌های هیستوگرام گرادیان‌های جهت دار و پیاده سازی تشخیص اشیا با استفاده از آن‌ها است.[20]

تشخیص اشیاء مبتنی بر یادگیری عمیق (به عنوان مثال، R-CNN، YOLO، SSD، CNN سریعتر)؛ این الگوریتم‌ها از شبکه‌های عصبی عمیق برای تشخیص اشیاء در تصاویر استفاده می‌کنند. آنها دقیق‌تر و دارند و می‌توانند طیف وسیعی از اشیاء را تشخیص دهند.[21]

برای استفاده از مدل‌های از پیش آموزش دیده برای این الگوریتم‌ها پشتیبانی می‌کند و پیاده سازی آنها را در برنامه‌ها آسان‌تر می‌کند. در مقایسه با الگوریتم‌های سنتی مانند HOG و Haar cascades رویکردهای مبتنی بر یادگیری عمیق به منابع محاسباتی بیشتری نیاز دارند اما عملکرد برتر را ارائه می‌دهند.[22,23]

HOG¹

تطبیق ویژگی:

شامل روش‌هایی برای تطبیق ویژگی‌ها می‌شود، مانند SIFT (تبديل ویژگی تغییرناپذیر مقیاس) و SURF (ویژگی‌های قوی با سرعت بالا). این روش‌ها برای کارهایی مانند هم‌ترازی تصویر و تشخیص اشیا مفید هستند، اما ممکن است برای تشخیص عمومی شی در صحنه‌های پیچیده مؤثر نباشند. به طور خلاصه، OpenCV طیف وسیعی از ابزارها و الگوریتم‌ها را برای تشخیص اشیا ارائه می‌کند، از روش‌های سنتی مانند آبشار HOG و Haar تا رویکردهای مبتنی بر یادگیری عمیق. انتخاب الگوریتم به عواملی مانند پیچیدگی کار تشخیص، منابع محاسباتی موجود و دقت مطلوب بستگی دارد.^[24,25]

۶-۵- نتیجه‌گیری

در این فصل به نحوه معادل‌سازی، معیارهای آن و فاکتورهای دخیل در این معادل‌سازی پرداختیم و در نهایت نتیجه آزمایش‌های ثبت‌شده را مورد بررسی قرار دادیم. در همه آزمایش‌ها تولید نقشه توسط طراح منابع بیشتری مصرف کرد.

فصل 6:

نتیجه‌گیری و کارهای آینده

۱-۶- مقدمه

راهکار ارائه شده را در فصل های قبل ارائه و سپس با معیارهای معرفی شده آن را مورد ارزیابی قرار دادیم. در این فصل ابتدا به بررسی کلی راهکار می پردازیم و سپس به معرفی کارهایی برای آینده خواهیم پرداخت.

۲-۶- نتیجه گیری

در این مقاله راهکاری ارائه شد تا بتوان نقشه‌ی اولیه بازی را که همواره طراحان تلاش میکنند با کمک کد نویسان در بازی های متفاوت پیاده‌سازی کنند با اسفاده از ابزاری به صورتی راحت تر، در زمانی کمتر و بدون دخالت و کمک کد نویسان پیاده‌سازی شود. در ادامه به بررسی راه هایی برای کارآمدی بیشتر این راهکار می‌پردازیم و هر کدام را به صورت خلاصه توضیح میدهیم.

۳-۶- کارهای آینده

از آنجایی که این راه حل ارائه شده دارای مزایا و معیایی بوده، برای بهبود عملکرد آن می توان این راهکار راه بهبود بخشید. در زیر به بررسی مواردی برای بهبود این راهکار می پردازیم.

بهبود دقت برنامه :

یکی از مواردی که باعث سخت شدن استفاده از این برنامه میشود دقت الگوریتم در تشخیص شی های خواسته شده است. در مواردی که نقشه‌ی اولیه با استفاده از دست کشیده و اسکن شده باشد ممکن است خطاهایی هنگام کشیدن رخ دهد (مانند: ضخامت که ممکن است در تمام وجهه های شکل کشیده شده یکسان نباشد و یا زاویه که ممکن است در مواردی مانند کشیدن مربع و یا مکعب حتماً ۹۰ درجه نیاشد) که این امر تشخیص شکل را با استفاده از شکل های از قبل ذخیره شده سخت میکند. در این موقع میتوان با تغییر الگوریتم تشخیص شی با بعضی از الگوریتم های گفته شده در فصل قبل و یا تغییر درصد خطای ابزار تشخیص شی این امر را بهبود بخشید. در حال حاضر امکان استفاده از اشکال از پیش ذخیره شده داده شده و در صورت نیاز طراح میتواند با دادن الگوی خود به برنامه شی مورد نظر خود را تشخیص دهد.

تشخیص عمق :

در روش فعلی برای نقشه های سه بعدی عمق همیشه برابر ۰ در نظر گرفته میشود و اجسام سه بعدی در یک لول قرار میگیرند. یکی از راه های برطرف کردن این مشکل در نظر گرفتن سطح های مختلف برای قالب های قرار داده شده در برنامه است که طراح میتواند بگوید جسم گفته شده در کدام سطح قرار بگیرد. راه دیگر به این صورت است که عکس اسکن شده از نقشه شامل سایه روشن هایی باشد که با استفاده از درصد تیرگی قالب اجسام بالا تر یا پایین تر قرار داده شوند (برای مثال جسمی که رنگ آن مشکی است در عمق ۶ و جسم سفید در عمق ۱ قرار داده شود).

مراجع

-
- [1] World of Level Design. (n.d.). Creating 3D Maps in Google Maps. Retrieved from https://www.worldofleveldesign.com/categories/level_design_tutorials/google-maps-3d.php access date: 16 Feb 2024
 - [2] Hyperallergic. (2017, November 9). This Medieval Fantasy City Generator Can Help You Dive Into Your Imagination. Retrieved from <https://hyperallergic.com/413148/medieval-fantasy-city-generator/> access date: 16 Feb 2024
 - [3] Python GUIs. (n.d.). Create Buttons in Tkinter. Retrieved from <https://www.pythonguis.com/tutorials/create-buttons-in-tkinter/> access date: 16 Feb 2024
 - [4] Analytics Vidhya. (2022, August). Object Detection Lite: Template Matching. Retrieved from <https://www.analyticsvidhya.com/blog/2022/08/object-detection-lite-template-matching/> access date: 16 Feb 2024
 - [5] PyImageSearch. (2021, March 29). Multi-Template Matching with OpenCV. Retrieved from <https://pyimagesearch.com/2021/03/29/multi-template-matching-with-opencv/> access date: 16 Feb 2024
 - [6] PyImageSearch. (2014, November 17). Non-Maximum Suppression for Object Detection in Python. Retrieved from <https://pyimagesearch.com/2014/11/17/non-maximum-suppression-object-detection-python/> (accessed February 16, 2024).
 - [7] PyImageSearch. (2015, January 26). Multi-Scale Template Matching Using Python and OpenCV. Retrieved from <https://pyimagesearch.com/2015/01/26/multi-scale-template-matching-using-python-opencv/> (accessed February 16, 2024).
 - [8] MathWorks. (n.d.). Convolutional Neural Network. Retrieved from [https://www.mathworks.com/discovery/convolutional-neural-network.html#:~:text=A%20convolutional%20neural%20network%20\(CNN,%2Dseries%2C%20and%20signal%20data.](https://www.mathworks.com/discovery/convolutional-neural-network.html#:~:text=A%20convolutional%20neural%20network%20(CNN,%2Dseries%2C%20and%20signal%20data.) (accessed February 16, 2024).
 - [9] BuiltIn. (n.d.). MobileNet. Retrieved from <https://builtin.com/machine-learning/mobilenet> (accessed February 16, 2024).
 - [10] AI TechSystems. (n.d.). 3D Shape Detection System. Retrieved from <https://medium.com/ai-techsystems/3d-shape-detection-system-d7e34286a2b1> (accessed February 16, 2024).
 - [11] Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). A Neural Algorithm of Artistic Style. *arXiv preprint arXiv:1508.06576*.
 - [12] Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. (2015)
 - [13] Cheng, Z., Yang, Q., Sheng, B.: Deep colorization. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 415–423

- [14] Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. CVPR (2015)
- [15] Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Advances in Neural Information Processing Systems. (2014) 2366–2374
- [16] Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 2650–2658
- [17] Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *European Conference on Computer Vision (ECCV)* (p.2).
- [18] Shaker, N., Togelius, J., Nelson, M.J.: Procedural Content Generation in Games: A Textbook and an Overview of Current Research. *Springer* (2016)
- [19] Viola, P., & Jones, M. J. (2001). Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001.
- [20] Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2005.
- [21] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems, 28
- [22] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2016.
- [23] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., & Reed, S. (2016). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision. ECCV 2016.
- [24] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2), 91–110.
- [25] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded-Up Robust Features. European Conference on Computer Vision. ECCV 2006.

واژه نامه

واژه‌نامه فارسی به انگلیسی

level generation.....	نقشه سازی
image processing.....	پردازش تصویر.
scene.....	صحنه
component.....	عنصر
template matching.....	تطبیق قالب
level design.....	طراحی نقشه
user interface.....	رابط کاربری گرافیکی
Non-Maximum suppression.....	سرکوب غیر حداقلی
bounding box.....	جعبه مرزی
multi scale template matching.....	تطبیق قالب چند مقیاسی
pyramid approach.....	رویکرد هرمی
library.....	کتابخانه
application programming interface.....	رابط برنامه نویسی کاربردی
convolution	پیچیدگی
pooling.....	ادغام
corrected linear unit.....	واحد خطی اصلاح شده
neural network.....	شبکه عصبی
time series.....	سری زمانی
deep complexity.....	پیچیدگی عمیق
fine-tune model.....	مدل دقیق
transfer learning.....	یادگیری انتقالی
style transfer.....	انتقال سبک
content loss	از دست دادن محتوا
edge.....	لبه
overlapping	همپوشانی
computer vision.....	بینایی کامپیوتری
bias.....	تعصبات

Abstract

Initial mapping is a very basic step in the production of a game map by a game designer who creates the most basic game map, one of the most important steps is to create a simple overview of the map that the game designers will help you to Do design ideas and patterns. Expressing the core of the game graphically, converting the basic map maker to a scene in Unity may have its own challenges and difficulties for an open designer, such as: preparing files, using Unity) creating a new project in Unity, building A new scene and..., creating game components (using the Unity editor to create game components and placing them at appropriate times based on the map and...) In this thesis, we are going to produce a tool to generate three Next, or two, we decide on the map drawn by the designers in Unity, with the aim of simplifying these issues for the game designers and helping them eliminate the things they don't need .Working with special elements of Unity (collider component) and coding creates its initial scene.

and receives the scanned image and recognizes the components of the map that are simulated in a simple way (such as: circle, triangle, rectangle, etc.) and converts it into a data object for the program, then in the creation scene has been

put Image processing in this project is done using the opencv tool and one of its algorithms called TM_CCOEFF_NORMED, which is one of the methods adapted for the format. The matching method is a technique used in computer vision to find a sub-image (pattern) in an image.

The image is designed. By using the tkinter library, the use of this tool has become easier for users. The output of this program is a two-dimensional or three-dimensional file (select the user) which is converted into the desired scene or game by placing this file in Unity and after performing a series of processes.



**Iran University of Science and Technology
School of Computer Engineering**

**Implementation of the tool to automatically convert the
sample image to the stage ready for the Unity game engine.**

**A Thesis Submitted in Partial Fulfillment of the Requirement for the Bachelor
Degree of
Computer Engineering**

**By:
Hana Kariman**

**Supervisor:
Dr. Behrooz Minaei**

February 2024