**بسمه‌تعالی**

عنوان:

**گزارش نهایی پروژه نهایی**

توسط:

**مجتبی هاشمی**

درس:

**ربات‌های متحرک خودگردان**

استاد:

**دکتر محمد‌علی کیوان‌راد – دکتر مهدی جوانمردی – دکتر خادمیان**

سال تحصیلی 1403-1404

|  |  |  |
| --- | --- | --- |
|  | **فهرست مطالب** |  |

[فصل 1: مقدمه 5](#_Toc190100795)

[فصل 2: ناوبری و الگوریتم جست‌و‌جو 6](#_Toc190100796)

[2‌-1‌ الگوریتم جست‌وجو و پوشش کامل 6](#_Toc190100797)

[2‌-2‌ ناوبری 9](#_Toc190100798)

[2‌-3‌ جلوگیری از برخورد در ربات متحرک خودگردان 11](#_Toc190100799)

[فصل 3: استفاده از مدل YOLOv8 در پروژه تشخیص اشیاء با تصاویر پهپاد 13](#_Toc190100800)

[3‌-1‌ معرفی مدل YOLO 13](#_Toc190100801)

[فصل 4: مکان یابی در سیستم جهانی 17](#_Toc190100802)

[4‌-1‌ بارگذاری و استفاده از داده‌های کالیبراسیون 17](#_Toc190100803)

[4‌-2‌ اصلاح تصاویر 18](#_Toc190100804)

[4‌-3‌ استخراج پارامترهای خارجی دوربین 20](#_Toc190100805)

[4‌-4‌ تبدیل مختصات پیکسلی به مختصات جهانی 21](#_Toc190100806)

[4‌-5‌ نکات مهم در تبدیل سیستم‌های مختصاتی 21](#_Toc190100807)

[4‌-6‌ روش حل چالش بازشناسی اشیاء 23](#_Toc190100808)

|  |  |  |
| --- | --- | --- |
|  | **فهرست شکل‌ها** |  |

[شکل 2- 1 نمونه ای از حرکت زیگزاگی پهباد 6](#_Toc190100809)

[شکل 2- 2 نمونه تصویر از حرکت در محیط سوله 8](#_Toc190100810)

[شکل 2- 3 مسیر حرکت و ارتفاع در محیط سربسته 8](#_Toc190100811)

[شکل 3- 1 نمونه ای از تصاویر دیتاست visdron 14](#_Toc190100812)

[شکل 3- 2 نمونه‌ای از تصاویر لیبل زده شده بر روی شبیه ساز 15](#_Toc190100813)

[شکل 3- 3 نتایج آموزش و فاین تیونینگ بر روی داده‌های برچسب‌گذاری شده 15](#_Toc190100814)

[شکل 4- 1 نمونه تصویر شطرنجی محیط شبیه‌ساز 17](#_Toc190100815)

[شکل 4- 2 تشخیص گوشه‌های داخلی صفحه شطرنجی 17](#_Toc190100816)

[شکل 4- 3 نمونه تصویر اصلاح شده 18](#_Toc190100817)

[شکل 4- 4 تصویر عمق و بازه مربوطه برای عمق سنجی 19](#_Toc190100818)

[شکل 4- 5 تصویر رنگی موردنظر برای شناسایی اشیاء 19](#_Toc190100819)

[شکل 4- 6 تصویر اصلاح شده و همان بازه 19](#_Toc190100820)

[شکل 4- 7 خروجی اشیاء تشخیص داده شده به همراه موقعیت مکانی آن‌ها روی نقشه 22](#_Toc190100821)

# مقدمه

این پروژه با هدف طراحی و پیاده‌سازی یک سیستم هوشمند برای جستجو، شناسایی و مکان‌یابی اشیاء در محیط فرودگاه با استفاده از یک پرنده خودگردان (ربات متحرک) در محیط شبیه‌سازی شده انجام می‌شود. پرنده مورد نظر مجهز به دوربینهای RGB و عمق است و با بهره‌گیری از الگوریتمهای پردازش تصویر، ناوبری و تبدیلات هندسی، وظیفه دارد تمامی خودروها، افراد و اشیاء تعریف‌شده در محدوده جادههای پیرامونی فرودگاه را شناسایی و موقعیت سه‌بعدی آنها را در دستگاه مختصات جهانی گزارش کند. این پروژه در مرز میان رباتیک، بینایی ماشین و هوش مصنوعی قرار دارد و چالش‌های فنی متعددی را در زمینه‌های کالیبراسیون سخت‌افزار، تبدیلات مختصات، استراتژی جستجو و شناسایی اشیاء پوشش می‌دهد.

# ناوبری و الگوریتم جست‌و‌جو

## الگوریتم جست‌وجو و پوشش کامل

1. حرکت مستقیم و بررسی محیط

پهپاد به صورت خودکار در مسیر مستقیم به سمت جلو حرکت می‌کند و در هر لحظه، از طریق سنسورهای عمق و دوربین‌های نصب‌شده، محیط اطراف خود را برای تشخیص موانع بررسی می‌کند. در صورتی که مانعی شناسایی شود، پهپاد به صورت خودکار ارتفاع خود را تنظیم می‌کند تا فاصله ایمن از مانع حفظ شود و بدون هیچ نوع برخوردی به مسیر خود ادامه دهد. این فرآیند به صورت پیوسته تکرار می‌شود تا پهپاد به انتهای مسیر مستقیم برسد.

1. الگوریتم حرکت زیگزاگی

پس از رسیدن به انتهای مسیر مستقیم، پهپاد جهت حرکت خود را به سمت پایین تغییر می‌دهد و با سرعت ثابت 10 متر بر ثانیه، به مدت 6 ثانیه در جهت مخالف حرکت می‌کند. سپس، پهپاد مسیر خود را مجدداً به سمت جلو تنظیم می‌کند و به حرکت مستقیم ادامه می‌دهد تا به انتهای مسیر برسد. این الگوریتم حرکتی به صورت مداوم تکرار می‌شود و باعث می‌شود پهپاد به طور مرحله‌ای به سمت پایین‌ترین نقطه یا خروجی حرکت کند.

با استفاده از این الگوریتم حرکت زیگزاگی، پهپاد قادر است به طور کامل محیط را پوشش دهد و از هرگونه برخورد با موانع احتمالی جلوگیری کند. این رویکرد به عملیات شناسایی و بازرسی محیط اطمینان بیشتری اضافه می‌کند و آن را دقیق‌تر و ایمن‌تر می‌سازد.

|  |  |
| --- | --- |
|  |  |

نمونه ای از حرکت زیگزاگی پهباد

1. حرکت در محیط‌های بسته و محدود

در محیط‌های بسته و محدود مانند سوله‌ها، حرکت پهپاد نیازمند دقت و کنترل بیشتری است. در این شرایط، پهپاد حرکات کوچک‌تری انجام می‌دهد و ارتفاع خود را به صورت دقیق‌تر کنترل می‌کند. ارتفاع پهپاد در طول مسیر به طور مداوم تحت پایش قرار می‌گیرد و در صورت لزوم، به منظور حفظ فاصله ایمن (حدود نیم متر از موانع)، اصلاح می‌شود.در این روش، پهپاد به جای پوشش کلی فضای سوله، تنها در مسیر میانی آن حرکت می‌کند. این رویکرد به پهپاد اجازه می‌دهد تا با دقت بیشتری محیط را بررسی کند و همچنین از موانع احتمالی در اطراف دیوارها و گوشه‌ها آگاه شود.

**مراحل اصلی**

1. ورود به سوله و تعیین موقعیت: پروسه با ورود پهپاد به سوله شروع می‌شود. در این مرحله، پهپاد با استفاده از دوربین‌های نصب‌شده، تصاویری از دو طرف سوله (جهت yaw = 90 و yaw = 270) تهیه می‌کند. این تصاویر به پهپاد کمک می‌کنند تا موقعیت خود را در سوله و فاصله از دیوارها را تشخیص دهد.
2. حرکت به سمت مرکز سوله: پس از تعیین موقعیت، پهپاد به سمت مرکز سوله (yaw = 180) حرکت می‌کند تا به نزدیکی یکی از دیوارها برسد. در این نقطه، دوباره تصاویری از دو طرف گرفته می‌شوند تا پهپاد اطمینان حاصل کند که در خط میانی سوله قرار دارد و فاصله مناسبی از دیوارها حفظ می‌کند.
3. حرکت بین دیوارها: پس از رسیدن به دیوار، پهپاد دوباره تصاویری از دو طرف تهیه می‌کند و سپس به سمت دیوار مقابل (yaw = 0) حرکت می‌کند. این الگوریتم حرکت بین دیوارها به صورت مکرر تکرار می‌شود. در هر مرحله، پهپاد تصاویری از محیط اطراف خود گرفته و موقعیت خود را به روز می‌کند.
4. بازگشت به مرکز و پیشروی:پس از اتمام حرکت بین دیوارها، پهپاد به مرکز سوله بازمی‌گردد و به مقدار مشخصی (چند متر) به جلو حرکت می‌کند. سپس، الگوریتم تصویربرداری و حرکت بین دیوارها دوباره تکرار می‌شود. این فرآیند به صورت مداوم ادامه می‌یابد تا پهپاد کل فضای سوله را به طور کامل پوشش دهد.

مزایای این روش

* پوشش کامل محیط : با استفاده از این روش، پهپاد قادر است تمام نقاط سوله را بازرسی کند و از هیچ زاویه‌ای غافل نماند.
* اجتناب از موانع : حرکت متمرکز در مسیر میانی و استفاده از تصاویر دوربین‌ها به پهپاد این امکان را می‌دهد تا از برخورد با موانع احتمالی جلوگیری کند.
* دقت بالا : با توجه به اینکه پهپاد به طور مداوم تصاویری از محیط اطراف خود گرفته و موقعیت خود را به روز می‌کند، عملکرد آن دقیق و ایمن است.

در تصویر زیر موقعیت و مسیر حرکت این روش را مشاهده می‌کنید:



نمونه تصویر از حرکت در محیط سوله

|  |  |
| --- | --- |
|  |  |

مسیر حرکت و ارتفاع در محیط سربسته

1. تشخیص عمق و تصمیم‌گیری

پهپاد به صورت پیوسته به سمت جلو حرکت می‌کند تا به مانعی برسد. در این نقطه، پهپاد متوقف می‌شود و عمق مسیر روبروی خود را بررسی می‌کند: اگر عمق کافی باشد (برای مثال، تا 10 متر)، پهپاد به مسیر خود ادامه می‌دهد و پس از طی مسافت مشخص، جهت خود را تغییر می‌دهد. در غیر این صورت، پهپاد به نزدیکی مانع پیش می‌رود و سپس چرخش می‌کند تا مسیر جدیدی پیدا کند. این فرآیند به صورت مکرر تکرار می‌شود تا پهپاد به تدریج از سوله یا محیط محدود خارج شود.

بهبود حرکت داخل سوله

ورود و چرخش اولیه :

* پهپاد ابتدا به نقطه شروع (ورودی انبار) منتقل می‌شود.
* سپس به سمت دیوار چرخش می‌کند تا آماده حرکت سیستماتیک در انبار شود.

حرکت زیگزاگ :

* پهپاد در یک الگوی زیگزاگ حرکت می‌کند که این الگو به صورت سیستماتیک کل انبار را پوشش می‌دهد.
* هنگامی که به دیوار یا مانع نزدیک می‌شود، 90 درجه چرخش می‌کند و به ردیف بعدی می‌رود.

تنظیم ارتفاع :

* ارتفاع پهپاد به طور پویا تنظیم می‌شود تا از برخورد با موانع جلوگیری شود و دقت تشخیص اجسام بهبود یابد.
* اگر پهپاد خیلی دور از زمین باشد، ارتفاع آن کاهش می‌یابد و اگر خیلی نزدیک باشد، ارتفاع آن افزایش می‌یابد.

خروج از انبار :

* پس از پایان حرکت زیگزاگ و رسیدن به انتهای انبار، پهپاد به نقطه خروج منتقل می‌شود.

ذخیره مسیر حرکت :

* موقعیت‌های (x, y, z) پهپاد در طول حرکت ذخیره می‌شوند و در نهایت در یک فایل CSV ذخیره می‌شوند.

1. نقش الگوریتم در ایمنی و کارایی

با استفاده از این الگوریتم حرکت هوشمندانه، پهپاد به صورت ایمن و کارآمد در محیط‌های محدود حرکت می‌کند و از برخورد با موانع جلوگیری می‌کند. این ویژگی به ویژه در محیط‌های صنعتی و انبارها، که شامل موانع زیادی هستند، اهمیت بالایی دارد. این الگوریتم به پهپاد این امکان را می‌دهد که به طور کامل و دقیق محیط را پوشش دهد، بدون نیاز به مداخله دستی کاربر، و همچنین اطمینان حاصل کند که تمام نقاط مورد نظر بازرسی شده‌اند.

## ناوبری

در این قسمت توضیح می‌دهیم که چگونه با استفاده از توابع نوشته شده کل محوطه پوشش داده شده و سرچ می‌شود:

تابع left\_to\_right

این تابع به منظور خودکارسازی الگوی حرکت پهپاد در محیط و جمع‌آوری داده‌های تصویری طراحی شده است. این الگو شامل حرکت‌های سیستماتیک به سمت راست و چرخش‌ها برای اسکن محیط است. مراحل اصلی این تابع عبارتند از:

* چرخش به سمت راست : پهپاد با سرعت 30 درجه در ثانیه به مدت 3 ثانیه به سمت راست چرخیده و زاویه مشاهده خود را تنظیم می‌کند.
* گرفتن تصویر : دوربین پهپاد فعال می‌شود و یک تصویر از محیط فعلی ثبت می‌کند.
* بازیابی اطلاعات عمق : با استفاده از تصویر عمق، حداقل فاصله تا موانع محیطی تعیین می‌شود.
* حرکت به جلو : بر اساس اطلاعات عمق، پهپاد به جلو حرکت می‌کند تا به فاصله ایمن از موانع برسد.
* چرخش نهایی : پهپاد دوباره به سمت راست چرخیده و الگوی حرکت خود را تکمیل می‌کند.

این تابع به پهپاد این امکان را می‌دهد که به صورت خودکار در یک منطقه مشخص حرکت کند و داده‌های تصویری را جمع‌آوری کند.

2. تابع right\_to\_left

این تابع مشابه تابع left\_to\_right عمل می‌کند، اما با این تفاوت که الگوی حرکت آن از چرخش به سمت چپ شروع می‌شود و سپس به سمت راست ادامه می‌یابد. مراحل این تابع شامل:

* چرخش به سمت چپ : پهپاد با سرعت 30 درجه در ثانیه به مدت 3 ثانیه به سمت چپ چرخیده و زاویه مشاهده خود را تنظیم می‌کند.
* گرفتن تصویر : یک تصویر از محیط فعلی ثبت می‌شود.
* بازیابی اطلاعات عمق : حداقل فاصله تا موانع محیطی از تصویر عمق استخراج می‌شود.
* حرکت به جلو : پهپاد بر اساس اطلاعات عمق به جلو حرکت می‌کند.
* چرخش نهایی : پهپاد دوباره به سمت راست چرخیده و الگوی حرکت خود را تکمیل می‌کند.

این تابع برای اسکن محیط در جهت‌های مختلف قابل استفاده است.

3. تابع get\_depth

تابع get\_depth مسئول بازیابی اطلاعات عمق از تصاویر عمقی است که توسط دوربین پهپاد گرفته می‌شوند. این تابع شامل مراحل زیر است:

* همگام‌سازی با محیط شبیه‌سازی : اطمینان حاصل می‌شود که تصویر عمق دقیق و بدون اشکال است.
* برش تصویر عمق : تصویر عمق به مناطق کوچک‌تر تقسیم می‌شود تا فقط ناحیه مورد نظر بررسی شود.
* محاسبه حداقل عمق : از بین مقادیر عمق موجود در تصویر، کمترین مقدار (که نشان‌دهنده نزدیک‌ترین مانع است) استخراج می‌شود.

این تابع نقش مهمی در اجتناب از موانع و حرکت ایمن پهپاد دارد.

4. تابع control\_altitude

تابع control\_altitude به منظور کنترل دقیق ارتفاع پهپاد طراحی شده است. این تابع از الگوریتمی استفاده می‌کند که ارتفاع فعلی پهپاد را با ارتفاع هدف مقایسه کرده و ضرورت حرکت عمودی را تعیین می‌کند. مراحل این تابع شامل:

* بازیابی ارتفاع فعلی : ارتفاع فعلی پهپاد از سنسورهای داخلی یا GPS خوانده می‌شود.
* مقایسه با ارتفاع هدف : ارتفاع فعلی با ارتفاع هدف مقایسه می‌شود.
* اجرای حرکت عمودی : اگر اختلاف وجود داشته باشد، پهپاد دستور حرکت به بالا یا پایین را دریافت می‌کند تا به ارتفاع هدف برسد.

این تابع تضمین می‌کند که پهپاد در ارتفاع مشخصی حفظ شود و از تغییرات غیرمجاز ارتفاع جلوگیری کند.

## جلوگیری از برخورد در ربات متحرک خودگردان

در این پروژه، برای جلوگیری از برخورد پرنده با موانع در محیط فرودگاه، از داده‌های دوربین عمق استفاده می‌شود. با توجه به اینکه پرنده در حال حرکت به جلو است، نیازی به بررسی کل تصویر نیست و تنها بخش‌هایی از تصویر که در مسیر حرکت پرنده قرار دارند، مورد تحلیل قرار می‌گیرند. این رویکرد باعث کاهش حجم محاسبات و افزایش سرعت پردازش می‌شود.

**مراحل اجرایی جلوگیری از برخورد**

1. دریافت تصویر عمق: تصویر عمق از دوربین پرنده دریافت می‌شود. این تصویر حاوی اطلاعات فاصله هر پیکسل از دوربین تا اشیاء در محیط است. مقادیر عمق در بازه‌ای بین ۰ تا ۸۰ متر قرار دارند، که در آن ۰ متر نشان‌دهنده نزدیک‌ترین فاصله و ۸۰ متر نشان‌دهنده دورترین فاصله است.
2. انتخاب ناحیه مورد بررسی: با توجه به جهت حرکت پرنده (معمولاً به جلو در راستای محور X بدنه)، تنها بخش‌هایی از تصویر که در مسیر حرکت پرنده قرار دارند، انتخاب می‌شوند. این ناحیه معمولاً به شکل یک مستطیل در مرکز تصویر و با عرضی متناسب با عرض مسیر پرواز تعریف می‌شود.
3. محاسبه کمترین فاصله تا موانع: در ناحیه انتخاب‌شده، کمترین مقدار عمق (نزدیک‌ترین فاصله تا مانع) محاسبه می‌شود. اگر این مقدار از یک حد آستانه (مثلاً ۱۰ متر) کمتر باشد، به این معناست که مانعی در مسیر پرنده وجود دارد.
4. واکنش به مانع: در صورت تشخیص مانع، پرنده می‌تواند با یکی از روش‌های زیر از برخورد جلوگیری کند:
   1. تغییر ارتفاع: افزایش ارتفاع پرنده (مثلاً از ۱۴ متر به ۲۲ متر) برای عبور از بالای مانع.
   2. تغییر مسیر: چرخش پرنده به چپ یا راست برای دور زدن مانع.
   3. توقف موقت: کاهش سرعت یا توقف کامل برای بررسی دقیق‌تر محیط.

**سناریوهای عملیاتی**

1. افزایش ارتفاع: در شرایطی که مانع در مسیر مستقیم پرنده قرار دارد، ارتفاع پرنده افزایش می‌یابد (مثلاً از ۱۴ متر به ۲۲ متر). این افزایش ارتفاع به پرنده امکان می‌دهد تا از بالای مانع عبور کند و دید بهتری از محیط اطراف داشته باشد. پس از عبور از مانع، پرنده به ارتفاع اولیه بازمی‌گردد و مسیر اصلی خود را ادامه می‌دهد.
2. کاهش ارتفاع: اگر کمترین فاصله تا موانع در کل تصویر از حد آستانه کمتر باشد (مثلاً کمتر از ۵ متر)، پرنده ارتفاع خود را کاهش می‌دهد. در این حالت، موقعیت افقی پرنده (X, Y) ثابت می‌ماند و تنها ارتفاع (Z) کاهش می‌یابد. این کار به پرنده امکان می‌دهد تا موانع نزدیک به زمین (مانند دوچرخه یا موتورسیکلت) را با دقت بیشتری شناسایی کند.
3. استفاده از کل تصویر برای بررسی دقیق: در شرایطی که پرنده در ارتفاع پایین پرواز می‌کند، از کل تصویر عمق برای تشخیص موانع استفاده می‌شود. این رویکرد دید وسیع‌تری از محیط اطراف فراهم می‌کند و به پرنده کمک می‌کند تا موانع احتمالی را با دقت بیشتری شناسایی کند. با استفاده از این اطلاعات، پرنده می‌تواند تصمیمات بهینه‌تری برای ادامه مسیر یا تغییر آن بگیرد.

**مزایای روش پیشنهادی**

* سرعت بالا: با تمرکز بر نواحی خاصی از تصویر، حجم محاسبات کاهش می‌یابد و سیستم به صورت بلادرنگ عمل می‌کند.
* انعطاف‌پذیری: پرنده می‌تواند با توجه به شرایط محیطی، ارتفاع یا مسیر خود را تغییر دهد.
* دقت بالا: استفاده از داده‌های عمق و تحلیل دقیق نواحی تصویر، دقت تشخیص موانع را افزایش می‌دهد.
* کاهش برخوردها: با شناسایی به‌موقع موانع و تغییر مسیر یا ارتفاع، احتمال برخورد به حداقل می‌رسد.

# استفاده از مدل YOLOv8 در پروژه تشخیص اشیاء با تصاویر پهپاد

## معرفی مدل YOLO

مدل YOLO (You Only Look Once) به‌عنوان یکی از پیشرفته‌ترین روش‌های تشخیص اشیاء در تصاویر شناخته می‌شود. این الگوریتم، که بر اساس شبکه‌های عصبی عمیق ساخته شده است، توانسته است با سرعت و دقت بالا، اشیاء مختلف را در تصاویر شناسایی کند. YOLO از معماری منحصر‌به‌فردی استفاده می‌کند که تصویر را به سلول‌های کوچک تقسیم کرده و همزمان پیش‌بینی‌های مربوط به وجود اشیاء و موقعیت آن‌ها (جعبه‌های مرزی) را انجام می‌دهد.

ویژگی‌های برجسته YOLO:

* سرعت بالا: با استفاده از معماری یکپارچه، YOLO تصویر را به‌صورت یکجا وارد شبکه می‌کند و نتایج را در یک مرحله محاسبه می‌کند. این ویژگی آن را مناسب کاربردهای زمان واقعی می‌کند.
* قابلیت تعمیم‌پذیری: مدل‌های مبتنی بر YOLO قادرند به‌راحتی با تغییراتی مانند اندازه، نورپردازی و زاویه دید سازگار شوند و عملکرد قابل قبولی ارائه دهند.
* کاربردهای گسترده: این مدل در حوزه‌های مختلفی مانند نظارت هوایی، خودروهای خودران، رباتیک و تحلیل ویدئوهای ورزشی کاربرد دارد.

با وجود مزایای فراوان، YOLO با چالش‌هایی نیز روبروست، از جمله:

* کاهش دقت در تشخیص اشیاء کوچک.
* مشکلات مربوط به همپوشانی اشیاء.
* محدودیت در شناسایی اشیاء در شرایط نور کم.

استفاده از YOLOv8 بر روی داده‌های VisDrone

YOLOv8 آ نسخه‌ای از خانواده الگوریتم‌های YOLO است که با بهبودهای معماری و بهره‌گیری از تکنیک‌های پیشرفته یادگیری عمیق، توانایی شناسایی سریع و دقیق اشیاء را در محیط‌های پیچیده ارائه می‌دهد. در این پروژه، از مدل YOLOv8 به عنوان یک شبکه پیش‌آموزش‌دیده بر روی داده‌های VisDrone استفاده شده است.

دیتاست VisDrone:

مجموعه داده VisDrone شامل تصاویر هوایی متنوعی از محیط‌های شهری و روستایی با زوایای مختلف است. این دیتاست به‌دلیل چالش‌هایی مانند اشیاء کوچک، تغییرات نورپردازی و تفاوت‌های زاویه دید، به یک مرجع مهم برای آزمایش و ارزیابی الگوریتم‌های تشخیص اشیاء تبدیل شده است.



نمونه ای از تصاویر دیتاست visdron

مزیت استفاده از مدل پیش‌آموزش‌دیده:

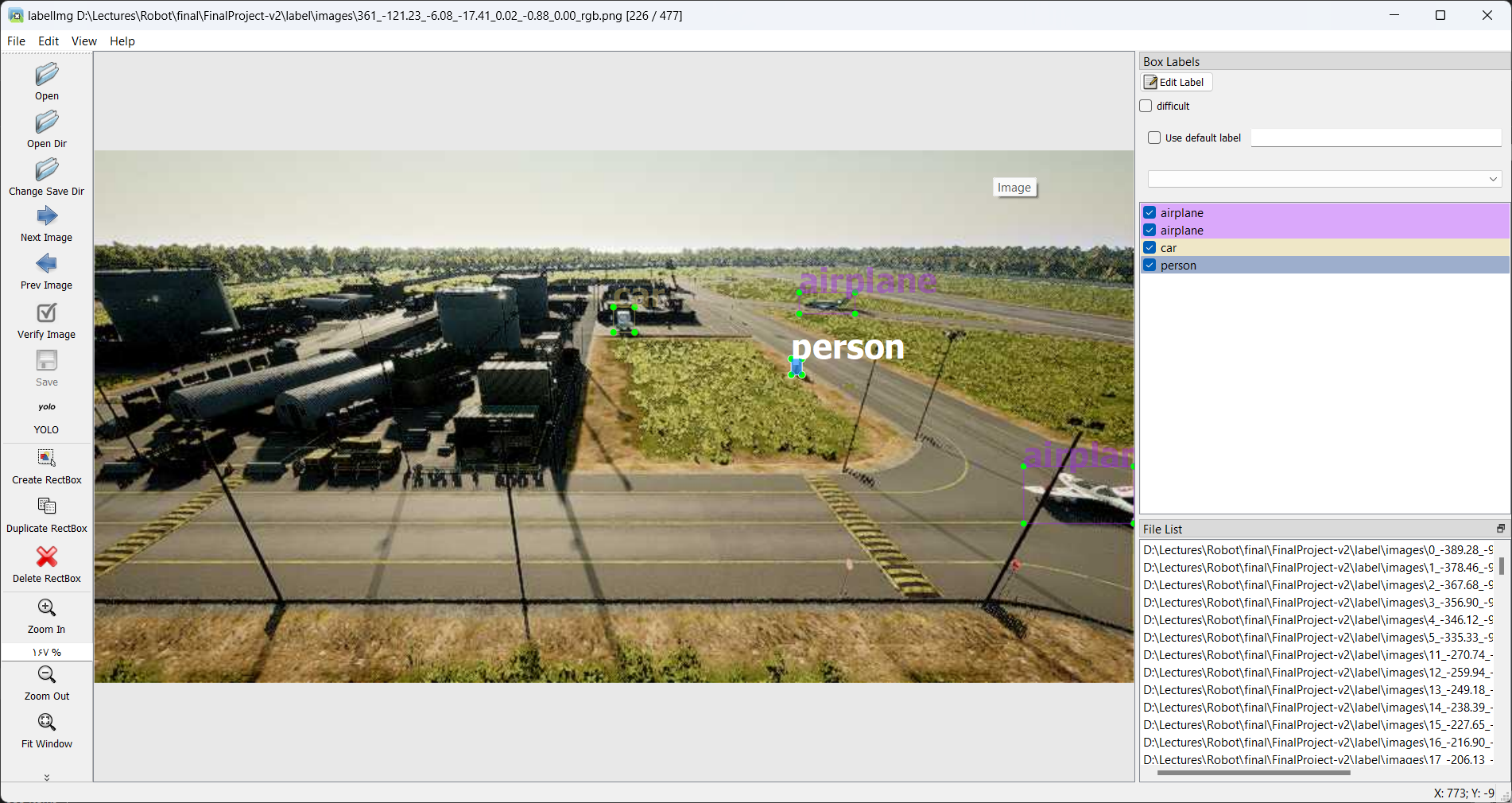
* کاهش زمان و منابع آموزش: مدل پیش‌آموزش‌دیده YOLOv8 دانشی اولیه از ویژگی‌های عمومی تصاویر دارد و تنها نیاز به تنظیم دقیق (فاین‌تیونینگ) روی داده‌های خاص پروژه دارد.
* افزایش دقت و تعمیم‌پذیری: با استفاده از دانش قبلی، مدل بهتر می‌تواند ویژگی‌های مهم اشیاء را استخراج کرده و از بروز مشکل بیش‌آموزی جلوگیری کند.

ساخت دیتاست محلی

در این پروژه، برای آموزش بهینه مدل، یک دیتاست سفارشی ساخته شده است که شامل حدود 500 تصویر گرفته‌شده از یک شبیه‌ساز می‌باشد. این تصاویر شرایط و محیط‌های متنوعی را شامل می‌شوند و به دلیل کنترل دقیق بر پارامترهای صحنه، امکان تنظیم دقیق ویژگی‌های مورد نیاز مدل فراهم شده است. تصاویر با کیفیت مناسب و در شرایط نوری متفاوت، زمینه‌ای مناسب برای آموزش شبکه در شناسایی دقیق اشیاء مختلف ایجاد می‌کنند.

برچسب‌گذاری دیتاست:

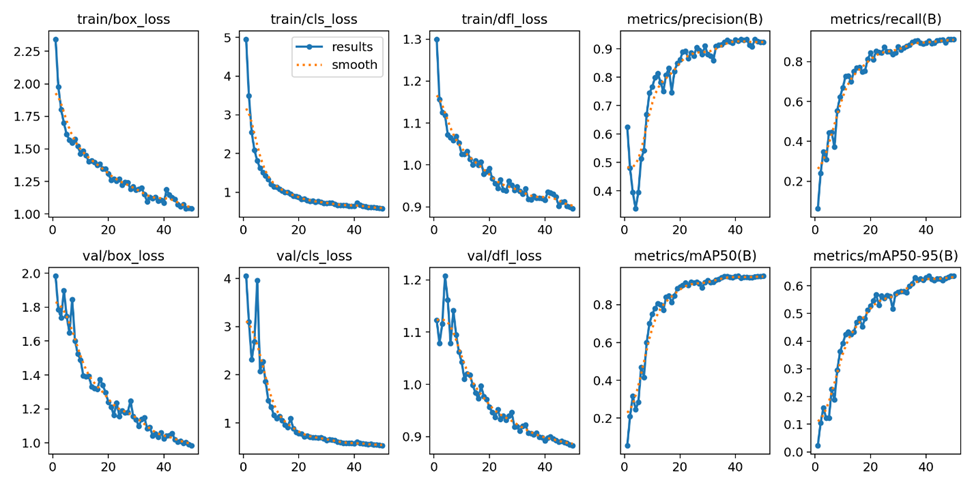
این دیتاست شامل حدود 1000 برچسب است که به دقت و با جزئیات روی چهار کلاس (شخص، ماشین، هواپیما و هلی‌کوپتر) تنظیم شده‌اند. این برچسب‌ها به مدل کمک می‌کنند تا با اطلاعات دقیق و منسجم درباره موقعیت و نوع هر شیء در تصاویر، فرایند یادگیری و شناسایی بهتری را انجام دهد.



نمونه‌ای از تصاویر لیبل زده شده بر روی شبیه ساز

نتایج آموزش مدل

پس از آموزش مدل، نتایج به دست آمده نشان می‌دهد که مدل توانسته است به خوبی وظیفه شناسایی اشیاء چهارکلاسه را به عهده بگیرد. دقت خروجی‌ها در شناسایی اشیاء مختلف و عملکرد قابل قبول شبکه در شرایط متنوع تصاویر شبیه‌سازی شده، از جمله نقاط قوت به‌دست آمده است.



نتایج آموزش و فاین تیونینگ بر روی داده‌های برچسب‌گذاری شده

روندهای آموزشی:

منحنی‌های آموزشی و ارزیابی حاکی از همگرایی مناسب مدل در طول دوره‌های آموزش بوده و کاهش خطا در مراحل نهایی نشان‌دهنده موفقیت در استخراج ویژگی‌های مهم از تصاویر است. افزایش تدریجی دقت در فازهای بعدی آموزش، شواهدی از تطبیق مناسب مدل با دیتاست ارائه شده را نشان می‌دهد.

چالش‌ها و پیشنهادات:

در برخی موارد، اشیاء کوچک یا مواردی با تغییرات زاویه دید خاص با دقت کمتری شناسایی می‌شوند. این موضوع می‌تواند نشان‌دهنده نیاز به بهبود‌های بیشتر در معماری شبکه یا افزایش تنوع دیتاست به منظور پوشش بهتر شرایط مختلف باشد. افزایش حجم داده‌ها و تنظیم دقیق پارامترهای مدل می‌تواند به دقت و قابلیت تعمیم بیشتری منجر شود.

تصاویری از مراحل آموزش روی دیتاست جدید:

|  |  |
| --- | --- |
|  |  |
|  |  |

نتایج فاین تیونینگ با دیتای بیشتر

# مکان یابی در سیستم جهانی[[1]](#footnote-1)

مراحل اصلی در این قسمت عبارت‌اند از:

## بارگذاری و استفاده از داده‌های کالیبراسیون

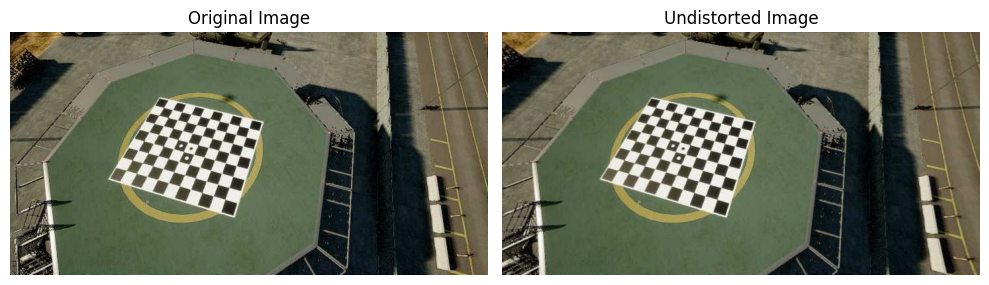
پیشتر درمورد نحوه کالیبراسیون در تمرین ها یادگرفته‌ایم در اینجا نیز با استفاده از صفحه شطرنجی موجود در شبیه‌ساز و جمع‌آوری تصاویر مربوطه و مراحل بدست آوردن داده های کالیبراسیون را مختصرا مشاهده می‌نمائیم:



نمونه تصویر شطرنجی محیط شبیه‌ساز



تشخیص گوشه‌های داخلی صفحه شطرنجی



نمونه تصویر اصلاح شده

و درنهایت ذخیره داده ‌های کالیبراسیون.

هدف: خواندن داده‌های کالیبراسیون از فایل (شامل ماتریس دوربین و ضرایب اعوجاج).

توضیح عملکرد:

* تابع «load\_calibration\_data» داده‌های ذخیره‌شده در فایل (مثلاً با فرمت NPZ) را بارگذاری کرده و در قالب یک دیکشنری ارائه می‌دهد.
* داده‌های بارگذاری‌شده شامل:
* camera\_matrix: ماتریس داخلی دوربین.
* dist\_coeffs: ضرایب اعوجاج.
* همچنین بردارهای چرخش (rvecs) و انتقال (tvecs) ممکن است موجود باشند.

## اصلاح تصاویر[[2]](#footnote-2)

هدف: حذف اعوجاج‌های ایجاد شده در تصاویر به واسطه‌ی لنز دوربین.

توضیح عملکرد:

* تابع «undistorted\_imgs» تصویر رنگی دریافتی (RGB) را به کمک ماتریس داخلی و ضرایب اعوجاج اصلاح می‌کند.
* این فرآیند تضمین می‌کند که مختصات پیکسلی به‌درستی منعکس‌کننده هندسه واقعی صحنه باشد.

در تصاویر زیر بررسی تطابق تصاویر عمق و تصویر رنگی با تصویر اصلاح شده را مشاهده می‌‎کنید:

|  |  |
| --- | --- |
| تصویر عمق و بازه مربوطه برای عمق سنجی | تصویر رنگی موردنظر برای شناسایی اشیاء |
| تصویر اصلاح شده و همان بازه | |

## استخراج پارامترهای خارجی دوربین[[3]](#footnote-3)

هدف: تعیین موقعیت و جهت‌گیری دوربین در فضای جهانی (NED) با استفاده از وضعیت پرنده.

توضیح عملکرد:

* دریافت وضعیت پرنده:
* موقعیت (position) و جهت‌گیری (orientation به صورت کواترنیون) پرنده دریافت می‌شود.
* تبدیل کواترنیون به ماتریس چرخش:
* کواترنیون دریافتی (با ترتیب درست) به ماتریس چرخش تبدیل می‌شود.
* اعمال آفست دوربین نسبت به بدنه:
* دوربین نسبت به بدنه پهپاد یک بردار آفست (مثلاً [0, 0, 0.24] متر) دارد.
* تصحیح جهت‌گیری دوربین:
* دوربین به اندازه زاویه پیچ (مثلاً 25 درجه رو به پایین) نسبت به بدنه چرخانده شده است.
* تبدیل سیستم‌های مختصاتی:
* یک ماتریس تبدیل ثابت از سیستم دوربین (که محورهای آن به صورت x: راست، y: پایین، z: جلو تعریف می‌شود) به سیستم بدنه NED (که محورهای آن x: جلو، y: راست، z: پایین است) تعریف می‌شود.
* به عنوان مثال، با نگاشت دستی محور‌ها می‌توان از ماتریس تبدیل زیر استفاده کرد:

ترکیب نهایی:

* ابتدا با ضرب ماتریسی (drone\_rot @ R\_cam\_to\_body @ R\_pitch) جهت‌گیری دوربین نسبت به بدنه محاسبه می‌شود.
* سپس جهت‌گیری نهایی دوربین در فضای جهانی از طریق ضرب ماتریسی با ماتریس چرخش پرنده به دست می‌آید.
* موقعیت دوربین در فضای جهانی با اضافه کردن آفست (بعد از اعمال چرخش پرنده) به موقعیت پرنده به دست می‌آید.

## تبدیل مختصات پیکسلی به مختصات جهانی[[4]](#footnote-4)

هدف: تبدیل مختصات یک نقطه (به عنوان مرکز یک جعبه تشخیص) در تصویر به مختصات سه‌بعدی در فضای جهانی.

توضیح عملکرد:

* استخراج مختصات پیکسلی:از مختصات جعبه تشخیص (bounding box) مرکز نقطه محاسبه می‌شود.
* اصلاح نقاط (Undistortion): در صورت لزوم، نقاط پیکسلی از اعوجاج تصویر اصلاح می‌شوند (اگرچه در این پروژه، تصویر اصلاح‌شده مستقیماً استفاده می‌شود).
* محاسبه عمق: از تصویر عمق، مقدار عمق در ناحیه کوچک اطراف مرکز جعبه استخراج می‌شود (مثلاً با استفاده از میانه مقادیر).
* محاسبه مختصات در سیستم دوربین: با استفاده از روابط مدل pinhole:

در اینجا (𝑢,𝑣) مرکز پیکسلی، (𝑐𝑥,𝑐𝑦) نقطه اصلی دوربین و 𝑓𝑥,𝑓𝑦 (فاصله کانونی) پارامترهای کالیبراسیون هستند.

* تبدیل به فضای جهانی: نقطه به دست آمده در فضای دوربین با ضرب در ماتریس جهت‌گیری دوربین (cam\_rot\_world) و اضافه کردن موقعیت دوربین (cam\_pos\_world) به فضای جهانی (NED) منتقل می‌شود.

## نکات مهم در تبدیل سیستم‌های مختصاتی

* سیستم دوربین: محورهای استاندارد دوربین معمولا به این صورت تعریف می‌شوند:

*Xcam : سمت راست*

*Ycam : سمت پایین*

*Zcam : سمت جلو*

* *سیستم جهانی (NED):*

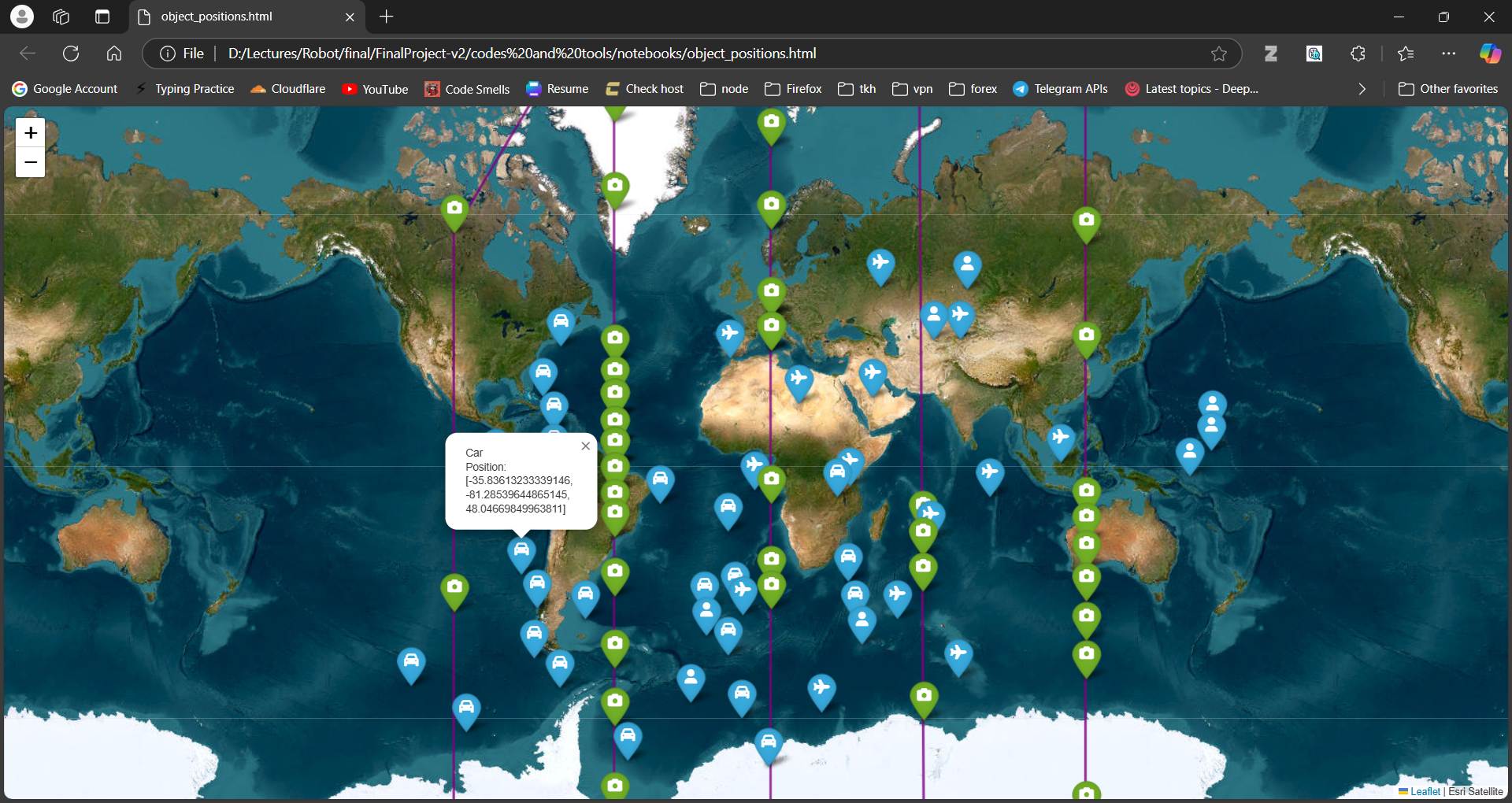
*NEDx : سمت جلو*

*NEDy : سمت راست*

*NEDz: سمت پایین*

*بنابراین، یک ماتریس تبدیل ثابت برای نگاشت بین این دو سیستم تعریف می‌شود (مثلاً همان ماتریس دستی با مقادیر که بیانگر تغییر ترتیب محور‌ها است).*

*در تصاویر زیر نتیجه نهایی از موقعیت دوربین و اشیاء شناسایی شده را مشاهده می‌کنید.*



خروجی اشیاء تشخیص داده شده به همراه موقعیت مکانی آن‌ها روی نقشه



## روش حل چالش بازشناسی اشیاء

در این روش، موقعیت تمام اشیا تشخیص داده شده در هر کلاس به طور جداگانه ذخیره می‌شود. هنگامی که یک شیء جدید تشخیص داده می‌شود، موقعیت آن با موقعیت تمام اشیا ذخیره شده در کلاس مربوطه مقایسه می‌شود.

اگر فاصله بین موقعیت شیء جدید و موقعیت نزدیک‌ترین شیء ذخیره شده از یک حد آستانه مشخص بیشتر باشد، شیء جدید به عنوان یک شیء جداگانه در نظر گرفته می‌شود و موقعیت آن به لیست اشیا ذخیره شده اضافه می‌شود. در غیر این صورت، شیء جدید به عنوان همان شیء قبلی در نظر گرفته می‌شود و از اضافه کردن دوباره آن به لیست جلوگیری می‌شود.

حد آستانه برای هر کلاس از اشیا متفاوت است و به طور تجربی تعیین می‌شود. به عنوان مثال، برای انسان‌ها این مقدار 0.5 متر، برای ماشین‌ها 10 متر و برای هواپیماها و هلیکوپترها 25 متر در نظر گرفته شده است.

با استفاده از این روش، می‌توان از تشخیص چندباره یک شیء جلوگیری کرد و از اطلاعات دقیق‌تری برای تصمیم‌گیری‌های بعدی استفاده کرد.

1. NED (North East Down) [↑](#footnote-ref-1)
2. Undistortion [↑](#footnote-ref-2)
3. Camera Extrinsic [↑](#footnote-ref-3)
4. Pixel-to-Global [↑](#footnote-ref-4)