دانشگاه آزاد اسلامی
واحد تهران جنوب
دانشکده فنی مهندسی

پایان نامه کارشناسی
مهندسی کامپیوتر– نرم افزار

عنوان:
شبیه‌ساز پرواز (Flight Simulator)

استاد راهنما:
دکتر امیر شهاب شهابی

نام و نام خانوادگی دانشجو:
وحید حیدری

شهریور ۱۳۹۲

با سپاس از خداوند مهربان، اساتید بزرگوار، دوستان عزیز و خانواه‌ام که مرا در به انجام رسانیدن این پروژه یاری رساندند.

فهرست مطالب

فهرست جداول

فهرست اشکال

چکیده

در این پایان‌نامه سعی بر ارائه ی نرم افزاری برای شبیه سازی پرواز می‌باشد. این شبیه‌ساز از تکنیک‌های مدل‌سازی و انتگرال‌گیری مبتنی بر روش‌های محاسبات عددی برای حل معادلات دیفرانسیل حرکت و پیشبردن شبیه‌سازی نسبت به زمان بهره می‌برند. همچنین برای شبیه‌سازی هر چه نزدیکتر به واقعیت، از مباحث مکانیک پرواز، آیرودینامیک، کنترل و پایداری و فیزیک حرکت اجسام صلب یا Rigid body استفاده شده و تمامی نیروها و گشتاورهای وارده به هواپیما محاسبه می‌شود.

در نهایت برای نمایش شبیه‌سازی از رابط برنامه نویسی DirectX استفاده شده تا تمام امور مربوط به گرافیک همانند رابط کاربر، نمایش مدل‌های سه بُعدی، انیمیشن، نمایش زمین و آسمان و غیره را به انجام برساند.

زبان برنامه نویسی انتخاب شده برای مدل‌سازی تمامی سیستم‌ها، کلاس‌ها، اشیاء و انجام محاسبات ریاضی، منطقی و فیزیکی در کامپیوتر، زبان C# می‌باشد. طبیعتاً با انتخاب معماری Client/Server و Net.، پلتفورم اجرای نرم افزار Microsoft Windows خواهد بود. متدولوژی توسعه‌ی نرم افزار RUP و ابزار مدل‌سازی نیز UML می باشد.

در فاز اول انجام این پروژه بیشتر تمرکز برروی مسائل مربوط به فیزیک و مکانیک پرواز و انجام تحقیقات و مدل‌سازی پارامترهای دخیل در پرواز هواپیماها و اجسام پرنده بوده است و مباحثی مانند بهینه‌سازی، نمایش زمین یا Terrain، مدیریت صحنه یا Scene managenent، برنامه نویسی صدا و

غیره در نظر گرفته نشده ولی تمام نیازمندی‌های نرم افزار به منظور توسعه‌های آینده در فصل نیازمندی‌ها ذکر شده است.

مقدمه

شبیه ساز پرواز برنامه‌ای است که می‌تواند پرواز هواپیما را به طور مصنوعی مدل‌سازی کند. از شبیه‌ساز پرواز می‌توان برای آموزش خلبانان هواپیماهای نظامی و غیر نظامی بهره برد. با توجه به هدف طراحی، یک شبیه‌ساز پرواز میتواند سخت افزار ، جزئیات مدل‌سازی و واقع گرایی متفاوتی داشته باشد. همچنین در یک شبیه‌ساز علاوه بر مدل‌سازی پرواز نیاز است تا محیط پرواز نیز شبیه‌سازی شود.

شکل ۱ – شبیه‌ساز کابین کنترل هواپیمای A380 Airbus ساخت Thales

امروزه استفاده از شبیه‌سازها در دو بخش نظامی و غیر نظامی برای اهداف آموزشی رو به گسترش است. این شبیه‌سازها به خدمه‌ی پرواز اجازه می‌دهند تا به راحتی تمام مانورهایی که حتی در حالت عادی برای انسان خطر آفرین هستند را بدون هیچ تهدیدی به اجرا در آورند. به عنوان مثال خلبانان نظامی قادرند مأموریت‌های بسیار پیچیده‌ای را که نیاز به مهارت بسیار بالایی دارد و اجرای آنها در شرایط واقعی یا زمان صلح غیر قابل قبول می‌باشد و یا استفاده از مهمات و تجهیزاتی که بسیار هزینه بر هستند انجام دهند. دلایل استفاده از شبیه‌سازها را می‌توان به صورت زیر ذکر کرد:

۱. **امنیت:** صنایع هوایی برپایه‌ی امنیت قرار دارد. اولین هدف صنایع هوایی این است که از امنیت تمام قسمت‌های هواپیما اطمینان حاصل کند؛ و این شامل آموزش‌ها نیز می‌شود.
۲. **مزایای اقتصادی:** هزینه‌ی آموزش از طریق شبیه‌ساز به طور معمول ۱۰ برابر کمتر از هزینه‌ی آموزش از طریق پرواز واقعی است. هزینه‌ی آموزش خلبانان برای یک شرکت هوایی می‌تواند سرسام آور باشد و آن را به مرحله‌ی ورشکستگی برساند. این امر می‌تواند موجب شود که خطوط هوایی برای کاهش هزینه‌ها و صرفه جویی در بودجه به آموزش از طریق شبیه‌ساز روی بیاورند.
۳. **مزایای آموزشی:** یکی از مزایای بسیار مهم شبیه‌سازها این است که زمانی که برای آموزش با آنها اختصاص داده می‌شود می‌تواند جایگزین زمان آموزش با هواپیما شود. اگر شبیه‌ساز به صورت کارآمدی ساخته شده باشد، آنگاه یک ساعت کار با شبیه‌ساز می‌تواند جایگزین یک ساعت، و یا حتی چند ساعت کار با هواپیما شود.
۴. **شبیه‌سازی مهندسی پرواز:** بیشتر شبیه‌ساز ها برای آموزش خلبانان استفاده می‌شوند. این درحالی است که شبیه‌سازها نقش مهمی در طراحی و ساخت هواپیماها نیز بازی می‌کنند. پر زحمت‌ترین بخش طراحی هواپیماهای مدرن امروزی را توسعه‌ی سیستم‌های هواپیما شامل می‌شود نه طراحی موتور یا سازه‌ی هواپیما. تا اوایل دهه ی ۱۹۸۰ آزمایشِ صحت عملکرد این سیستم‌ها تنها در جریان پرواز واقعی میسر بود. از کار افتادن این سیستم‌ها در این زمان بسیار گران تمام می‌شود و امنیت پرواز را به مخاطره می‌اندازد. در برنامه‌ی توسعه‌ی برخی از هواپیماهای جدیدِ امروزی از شبیه‌ساز مهندسی پرواز برای فراهم کردن بستر نرم افزار لازم برای پروازهای آزمایشی و تست‌های این بخش از توسعه بهره گرفته شده است.

شکل ۲ - یک دستگاه شبیه‌ساز پرواز آموزشی ساخت Thales

با پیشرفت میکرو کامپیوترها و به بازار عرضه شدن PCها ، کامپیوترهای PC برای اجرای برنامه‌های شبیه ساز به کار گرفته شدند. به دلیل آنکه پردازنده‌های این سیستم‌ها از توانایی بالایی برای انجام محاسبات ریاضی و فیزیک برخوردارند می‌توان به راحتی از آنها برای انجام محاسبات معادلات حرکت بهره گرفت.

فصل اول: نیازمندی ها

۱ – ۱ چشم انداز

قبل از ورود به نیازمندی‌ها باید چشم انداز و Vision پروژه بررسی شود تا هدف از انجام این پروژه روشن شود. انجام این پروژه در چندین فاز انجام خواهد شد. فاز اول شامل کلیه کارها و فعالیت‌هایی است که در این پایان‌نامه به انجام رسیده و نتایج حاصل و مستندات آن جمع آوری شده است. در فازهای بعدی به تکمیل بخش‌های دیگر نرم افزار پرداخته خواهد شد تا به یک نرم افزار کاربردی و تجاری قابل ارائه به بازار تبدیل شود.

۱ – ۲ Use caseها

نیازمندی‌ها که از آن با عنوان Use case نام برده می‌شود یکی از مهمترین بخش‌های هر پروژه است که تمام مراحل انجام پروژه تحت تاثیر آن قرار دارند. در ابتدا با مشخص کردن نیازمندی‌های پروژه، یک تقسیم بندی از فازهای مختلف توسعه نرم افزار نمایان می‌شود. برخی نیازمندی‌های انجام این پروژه در ادامه خواهد آمد.

۱ – ۳ موتور فیزیکی

تمرکز پایان‌نامه ارائه شده به انجام این بخش معطوف شده است و عمده فعالیت‌ها، تحقیقات، مستندات و برنامه‌ی تحویل داده شده به این قسمت اختصاص دارد. هدف از انجام این پایان‌نامه ارائه‌ی نرم افزاری برای شبیه‌سازی پرواز هواپیما بر پایه مدل‌سازی ریاضی از قوانین و معادلات حاکم بر حرکت اجسام پرنده در جوّ و محاسبه کارایی هواپیماها با توجه به پارامترهایی مانند قابلیت پروازی، برد، سقف پرواز، حداکثر سرعت، مداومت پروازی، اوج گیری، برخاست و فرود و قدرت مانوری می‌باشد. تمامی این مباحث ذکر شده موضوع علم مکانیک پرواز می‌باشد. این نیازمندی به قسمت‌های دیگری تقسیم می‌شود:

- شبیه‌سازی حرکت اجسام صلب

- تشخیص برخورد یا Collision detection
- پاسخگویی به برخورد یا Collision response
- انتگرال‌گیری از معادلات دیفرانسیل حرکت و حل آنها

- مدل‌سازی پرواز
  - مکانیک پرواز
  - آیرودینامیک
  - مشخصات کارایی وسایل پرنده‌ی در حال شبیه‌سازی
  - شبیه‌سازی کابین خلبان

۱ – ۴ موتور گرافیکی

از دیگر جنبه‌های بسیار مهم در شبیه‌سازی، گرافیک است و بیشترین Use caseها مربوط به گرافیک می‌شود یا به نوعی با آن ارتباط پیدا می‌کند. چون کُندترین، پرکارترین و پیچیده‌ترین قسمت شبیه‌ساز بخش گرافیک محسوب میشود. بنابر اهمیت این موضوع برای امور مربوط به گرافیک از رابط برنامه نویسی گرافکی Direct3D بهره برده شده است. به عقیده‌ی بسیاری Direct3D یکی از پیچیده‌ترین کتابخانه‌های گرافیکی است ولی در مقایسه با OpenGL امکانات بیشتری در اختیار برنامه نویسان گرافیک قرار می‌دهد.

برای سریع‌تر کردن برنامه نیاز به پیاده‌سازی یک الگوریتم بهینه‌سازی می‌باشد. بهینه‌سازی‌های گرافیکی مباحث پیچیده‌ای هستند که نیاز به تحلیل و بررسی دارند تا بهترین کارایی از نظر سخت افزاری و نرم افزاری حاصل شود. انتخاب یک الگوریتم مناسب از اهمیت ویژه‌ای برخوردار است. به دلیل این که شبیه‌سازی در محیط‌های باز و Out Door صورت می‌گیرد Field Of View بسیار وسیع است و تعداد اشیایی که باید در هر فریم برای کارت گرافیک ارسال شود زیاد است در نتیجه به شدت کارایی و سرعت اجرای برنامه پایین می‌آید. از مواردی که سرعت اجرا را بالا می‌برد پیاده‌سازی Frustum Culling است تا اشیایی که در زاویه دید نیستند برای کارت گرافیک ارسال نشوند چون در نهایت نمایش داده نخواهند شد. اما Frustum Culling به تنهایی کافی نیست و باید در کنار الگوریتم‌های دیگر مانند Quadtree به کار رود. همین طور الگوریتم‌هایی برای کاهش جزییات مدل‌هایی که در فواصل دور تر از دوربین قرار دارند نیاز است. اجسامی که مثلاً در فاصله‌ی ۱۰ کیلومتری از دوربین قرار گرفته‌اند نیازی ندارند که با جزئیات بالا نشان داده شوند ولی اجسامی که در فاصله ۵ متری از دوربین قرار گرفته‌اند باید با حداکثر جزئیات نمایش

داده شوند. الگوریتم‌هایی برای تغییر و تنظیم Level Of Detail وجود دارد. با این توضیحات بخش گرافیک نیز به قسمت‌های زیر تقیسم می‌شود:

- ایجاد Graphic device
  - ایجاد بهترین Device ممکن با توجه به مشخصات کارت گرافیک برای نمایش‌های گرافیکی
  - ایجاد و تغییر مشخصات Device با توجه به تنظیمات کاربر
- اداره کردنِ Device
  - اداره کردنِ Lost device ها
  - اداره کردنِ Resize
  - ازبین بردنِ Device ایجاد شده در انتهای برنامه به منظور آزاد سازی حافظه و کارت گرافیک
  - تغییر تنظیاتِ Device توسط کاربر و بازنشانی و اعمال آنها
- بهینه‌سازی‌های گرافیکی
  - بهینه‌سازی‌های مربوط به Terrain برای نمایش Real time
  - بهینه‌سازی‌های مربوط به نمایش مدل‌های سه بعدی
  - Frustum culling و Quadtree و الگوریتم‌های Scene management
- Digital Elevation Modeling برای نمایش عوارض و پستی بلندی‌های زمین
- GUI
- Visual effect
  - انفجار
  - دود
  - آب (دریا، اقیانوس)
  - ابر و مکعب آسمان (Sky box)
  - سایه‌ها با بهره گیری از تکنیک Shadow volume
  - درخشش لنز یا Lens flare
  - نورپردازی
  - High Level Shading Language

۱ – ۵ موتور صدا

با توجه به استفاده از DirectX SDK برای توسعه‌ی این نرم افزار، از جمله امکاناتی که این بسته در اختیار برنامه نویسان قرار می‌دهد امکان برنامه نویسی صدا می‌باشد. با استفاده از DirectSound می‌توان صدا را از دستگاه‌های ورودی مانند کارت صدا ضبط کرد و از طریق انواع مختلفی از دستگاه‌های خروجی پخش کرد. از ویژگی‌های بسیار پیشرفته و مفید این کتابخانه امکان پخش سه بعدی صدا یا 3dimentional positioning effect می‌باشد. هم چنین می‌توان انواع افکت‌های صوتی را روی فایل‌های صوتی با فرمت wav. به اجرا گذاشت. این کتابخانه با فراهم کردن APIهایی برای کار با Bufferهای کارت صدا به برنامه نویسان صدا اجازه می‌دهد تا به آسانی به کار با صداها بپردازند و با اجرای صداها و افکت‌های صوتی مناسب فضا را به واقعیت نزدیکتر کنند و محیط را به صورت چند رسانه‌ای در بیاورند تا مخاطب به شکل بهتری با برنامه تعامل داشته باشد. در واقع این بخش وظیفه‌ی Interactive Audio Programming را بر عهده دارد.

۱ – ۶ اداره کردنِ ورودی هایِ کاربر

برای پردازش ورودی‌های کاربر مانند Keyboard، Mouse، Joystick نیز DirectX یک کتابخانه به نام DirectInput را برای این منظور فراهم آورده است. این کتابخانه امکانات وسیعی برای شناسایی و خواندن انواع مختلفی از سخت افزارهایی که به عنوان ورودی به کار می‌روند فراهم می‌کند. برنامه نویس می‌تواند با استفاده از واسطهای آن، با کارایی بسیار بالایی با ابزار های ورودی/خروجی ارتباط برقرار کند و اطلاعات تولید شده را از روی آنها بخواند. قسمت‌های مختلف این بخش به قرار زیر است:

- شناسایی سخت افزارهای ورودی متصل شده به سیستم
  - Keyboard
  - Mouse
  - Joystick
- Input device listener

۱ – ۷ مدیریت حافظه

برای مدیریت منابع سیستم نیاز به طراحی الگوریتم‌های مدیریت تخصیص حافظه به صورت دینامیک هستیم. به عنوان مثال برای نمایش زمین نیاز به یک Object Pool داریم تا قطعات مختلف زمین که

متعلق به مناطق مختلف جغرافیایی می‌شوند را در موقع لزوم بارگیری کند تا هرگاه نیاز به نمایش آنها بود از داخل pool به آنها دسترسی سریع‌تری داشته باشیم و هر بار نیاز به بارگیری مجدد آنها نباشد. به این ترتیب سرعت و کارایی برنامه نیز افزایش می‌یابد. همچنین زمانی که نیاز به برخی از اشیاع نیست آنها را از pool خارج کند و اشیاء ضروری‌تر را در داخل آن بارگیری کند.

همچنین نیاز به طراحی سیستمی برای مدیریت ریسمان‌های (Thread) مختلف می‌باشد. به دلیل اینکه محاسبات مربوط به موتور فیزیکی بسیار سنگین و زمانبر هستند و برای هر جسمی که می‌خواهیم شبیه‌سازی بر روی آن انجام شود باید این محاسبات تکرار شوند، در نتیجه بهتر است این محاسبات در ریسمان‌های جداگانه انجام شوند و به ازای هر جسم یک ریسمان اختصاص یابد و در انتهای شبیه‌سازی نیز حافظه تخصیص داده شده آزاد سازی شود.

۱ – ۸ مدیریت پرونده ها

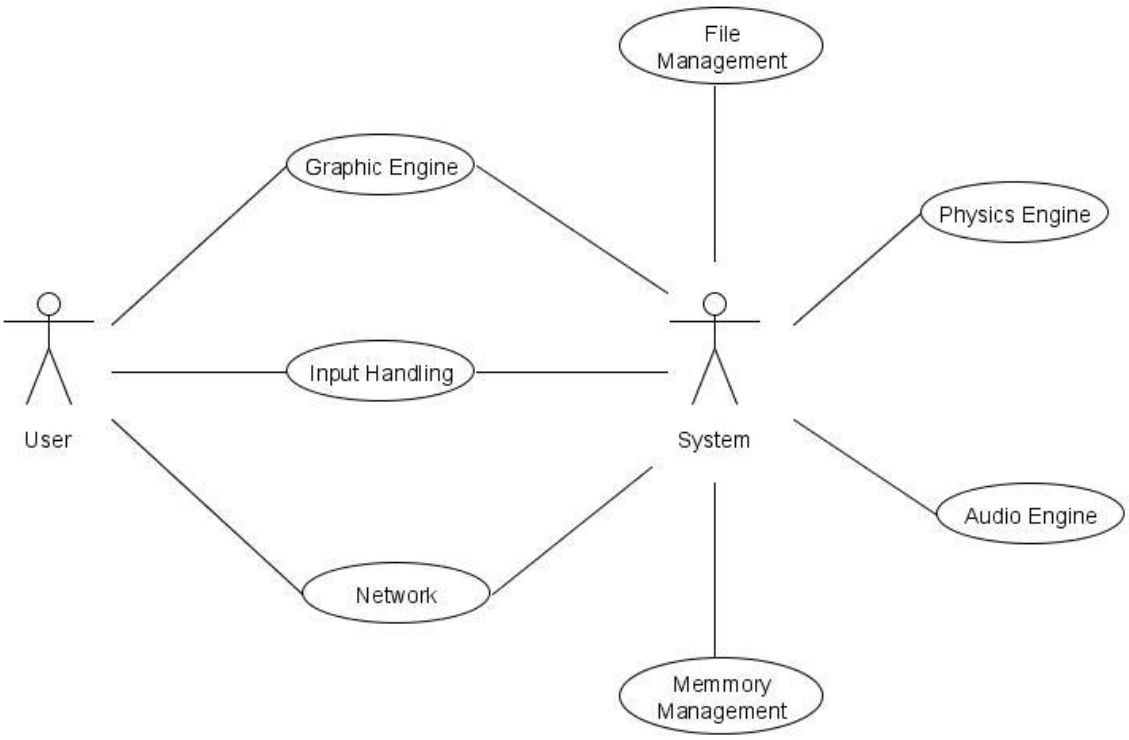
یک فرمت فایل مناسب برای نگهداری و ذخیره سازی تمامی اطلاعات مورد نیاز برنامه مانند تنظیمات، اطلاعات فیزیکی و کارایی مربوط به اجسام پرنده، پویانمایی‌ها و مدل‌ها باید طراحی شود. الگوریتمی برای فشرده‌سازی و فایل مجازی باید پیاده‌سازی شود. نیازمندی‌های این بخش به شرح زیر است:

- طراحی فرمت فایل مناسب
- فشرده‌سازی فایل‌ها
- فایل مجازی یا Virtual Directory

۱ – ۹ شبکه

برقراری ارتباط به صورت چند نفره برای انجام آموزش‌هایی که اجرای آن مأموریت‌ها باید با پروازهای چند فروندی انجام شود نیاز بستر شبکه دارد تا از طریق آن این ارتباط فراهم شود. به این منظور DirectX کتابخانه ی DirectPlay را فراهم کرده است.

در نهایت به صورت مختصر و اجمالی، Use caseها به شکل زیر خواهند بود:



شکل ۳ - Use case

فصل دوم: مدل سازی فیزیکی

۲ – ۱ معادلات حرکت

بنابر قوانین نیوتن هر حرکتی به محرک نیاز دارد. اعمال نیروهای مختلف به جسم سبب حرکت آن می‌شود. جهت این حرکت، همان جهت برایند برداری تمام نیرو های وارد بر جسم است. این نیروهای وارد شده به جسم هستند که برایندشان جهت، سرعت و شتاب حرکت را تعیین می کنند. معروف ترین قانون نیوتن، قانون دوم نیوتن در باره حرکت است که به صورت زیر می‌باشد:

$$F = ma,$$

که F برایند نیرو های وارد بر جسم، m جرم جسم و a شتاب خطی وارد بر مرکز جرمِ جسم است. اگر اندکی معادله را باز چینی کنیم خواهیم داشت:

$$\frac{F}{m} = a.$$

از رابطه‌ی اخیر در خواهیم یافت که جرم جسم به عنوان یک عامل باز دارنده در مقابل حرکت جسم عمل می‌کند. مشخص است که با اعمال یک نیروی ثابت به جسم، با افزایش جرم آن جسم شتاب، آن کاهش می‌یابد. می‌توان گفت جسمی با جرم بیشتر در مقابل حرکت مقاومت بیشتری از خود نشان می‌دهد. و نیز جسمی با جرم کمتر از خودش در مقابل حرکت مقاومت کمتری نشان می‌دهد.

قانون دوم نیوتن بیان می‌کند که شتاب وارد بر جسم هم جهت با برایند نیروهای وارد بر جسم است و در نتیجه شتاب و نیرو به صورت کمیت‌های برداری هستند. در حالت کلی ممکن است به یک جسم در یک لحظه چندین نیرو وارد شود، و این بدان معناست که نیروی حاصل برابر جمع برداری تمام نیروهای وارد بر آن جسم است. در نتیجه می‌توان نوشت:

$$\sum F = ma,$$

که بیان کننده بردار شتاب است.

در مختصات سه بعدی، بردارهای شتاب و نیرو دارای سه مؤلفه‌ی x ، y ، z هستند. در این حالت مؤلفه‌های معادله حرکت به صورت زیر می‌شوند:

$$\sum F = m\, a_x,$$

$$\sum F = m\, a_y,$$

$$\sum F = m\, a_z.$$

یک تفسیر دیگر از قانون دوم نیوتن این است که مجموع تمام نیروهای عمل کننده روی جسم برابر نرخ تغییرات اندازه حرکت جسم در طول زمان است، که می‌شود مشتق اندازه حرکت بر حسب زمان. اندازه حرکت برابر است با سرعت ضرب در جرم، و از آنجایی که سرعت یک کمیت برداری است در نتیجه اندازه حرکت نیز برداری است:

$$G = m\, v,$$

که در آن G اندازه حرکت خطی جسم، m جرم جسم و v برابر سرعت مرکز جرم جسم است. تغییرات اندازه حرکت نسبت به زمان همان مشتق اندازه حرکت نسبت به زمان می‌شود، پس خواهیم داشت:

$$\frac{dG}{dt} = \frac{d}{dt}\,(mv),$$

و با فرض ثابت بودن جرم می‌توان نوشت:

$$\frac{dG}{dt} = m\frac{dv}{dt}.$$

واضح است که تغییرات سرعت نسبت به زما ن برابر است با شتاب، پس داریم:

$$\frac{dG}{dt} = ma,$$

$$\sum F = \frac{dG}{dt} = ma.$$

تا اینجا ما فقط در مورد حرکت انتقالی اجسام بحث کردیم و از حرکت دورانی آنها سخنی به میان نیاوردیم. در واقع در دنیای سه بعدی علاوه بر حرکت انتقالی، یک جسم دارای حرکت دورانی نیز هست و

در نتیجه به معادلات بیشتری برای توصیف کامل حرکت جسم نیاز داریم. به طور مشابه نیاز به فرمولی داریم که جمع تمام گشتاورهای وارد بر جسم را به نرخ تغییرات اندازه حرکت زاویه‌ای در طول زمان نسبت دهد، در نتیجه می‌نویسیم:

$$\sum M_{cg} = \frac{d}{dt}\,(H_{cg}),$$

که در آن $\sum M_{cg}$ برابر است با مجموع گشتاورهای وارد بر مرکز جرم جسم، و H برابر اندازه حرکت زاویه‌ای جسم است. مقدار M_{cg} را می‌توان به صورت زیر در نظر گرفت:

$$M_{cg} = r \times F,$$

که F در آن نیروی وارد بر جسم و r بردار فاصله از F ، عمود بر خط عمل کننده‌ی نیروی F تا مرکز جرمِ جسم می‌باشد، و × نیز ضرب خارجی می‌باشد.

اندازه حرکت زاویه‌ای برابر با مجموع گشتاورهای وارد بر جسم در حول محور دوران است، که فرض می‌کنیم از مرکز جرم می‌گذرد. آن را به این صورت می‌توان در نظر گرفت:

$$H_{cg} = \sum r_i \times m_i\,(\omega \times r_i),$$

که در آن i نشان دهنده i-امین ذره‌ای است که جسم را می‌سازد، $\boldsymbol{\omega}$ برابر سرعت زاویه‌ای جسم حول محور دوران می‌باشد و $(\omega \times r_i)$ برابر اندازه حرکت زاویه‌ای i-امین ذره‌ای است که اندازه آن برابر ωr_i می‌باشد . برای دوران حول محور مفروض داده شده می‌توان معادله را به صورت زیر باز نویسی کرد:

$$H_{cg} = \int \omega\, r^2\, dm,$$

که در آن $\int r^2\, dm$ برابر با گشتاور اینرسی یا I می‌باشد ، و در نتیجه با جایگذاری آن داریم:

$$H_{cg} = I\,\omega.$$

با مشتق گیری از معادله اخیر بر حسب زمان خواهیم داشت:

$$\frac{d\,H_{cg}}{dt} = \frac{d}{dt}\,(I\,\omega) = \frac{I\,d\omega}{dt} = I\alpha,$$

که در آن $\boldsymbol{\alpha}$ برابر شتاب زاویه‌ای جسم حول محور داده شده است. در نهایت می‌توان نوشت:

$$\sum M_{cg} = I\alpha.$$

به طور معمول M و $\boldsymbol{\alpha}$ کمیت‌های برداری هستند، در حالی که I می‌تواند یک تنسور (tensor) باشد، زیرا ممکن است گشتاور اینرسیِ حول محورهای دوران متفاوت باشد. تنسور یک عبارت ریاضی است که هم دارای مقدار است و هم دارای جهت ولی ممکن است در جهات مختلف یکسان نباشد. در اینجا گشتاور اینرسی را با تنسور نمایش می‌دهیم. در حالت کلی گشتاور اینرسی با یک ماتریس۳ × ۳ نمایش داده می‌شود.

در اینجا لازم به ذکر است که معادلات اخیر نسبت به مختصات جهانی یا global coordinate system می‌باشند، نه مختصات محلی جسم. این مختصات برای معادلات خطی حرکت مناسب هستند، زیرا می‌توان توسط آنها مکان جسم و سرعت جسم را نسبت به مختصات جهانی ردیابی کرد. اما از نظر محاسباتی، نمی خواهیم معادلات حرکت دورانی را هم به همین صورت محاسبه کنیم. زیرا گشتاور اینرسی، زمانی که نسبت به مختصات جهانی در نظر گرفته می‌شود، نسبت به مکان و جهت گیریِ جسم تغییر می‌کند. این بدان معناست که در طول شبیه‌سازی باید مقدار ماتریس اینرسی و معکوس آن را مکرراً محاسبه کنیم، که بسیار کارایی را پایین می‌آورد. بهتر است که معادله را نسبت به محورهای محلی جسم باز نویسی کنیم. زیرا در این صورت تنها یکبار محاسبه می‌شود و آن هم قبل از شروع شبیه‌سازی و در ابتدای آن.

در حالت کلی، مشتق بردار V نسبت به زمان در یک سیستم مختصات ثابت که دوران ندارد طبق معادلات زیر با سیستم مختصات دارای دوران ارتباط دارد:

$$(\frac{d\,V}{dt})_{fixed} = (\frac{d\,V}{dt})_{rotating} + (\,\omega \times V),$$

که عبارت $(\boldsymbol{\omega} \times \mathrm{V})$ نشان دهنده‌ی اختلاف بین مشتق V نسبت به زمان است که در سیستم مختصات ثابت اندازه گیری شده به مشتق V نسبت به زمان در سیستم مختصات دارای دوران. از این معادله می‌توان برای باز نویسی معادلات حرکت دورانی در سیستم مختصات محلیِ ثابتِ جسم استفاده کرد:

$$\sum M_{cg} = \frac{dH_{cg}}{dt} = I\left(\frac{d\omega}{dt}\right) + (\,\omega \times (I\omega)\,),$$

که در آن گشتاور اینرسی و سرعت دورانی نسبت به مختصات محلی هستند. گرچه ممکن است این معادله از آنچه قبلاً ارائه شده کمی پیچیده‌تر به نظر برسد ولی برای استفاده کردن آسان‌تر است زیرا I یک عبارت ثابت است و در طول شبیه‌سازی تنها زمانی تغییر می‌کند که هندسه یا جرم جسم تغییر کنند.

فصل سوم: مکانیک پرواز

۳ – ۱ نیروهای آیرودینامیکی

معمولاً دو نوع نیرو بر روی یک جسم در حال پرواز یکنواخت و بدون شتاب عمل می‌کنند، که عبارتند از نیروهای جسم و نیروهای سطح. نیروهای جسمی از یک فاصله به جسم اثر می‌کنند.

در مورد هواپیماها این نیروها، نیروهای جاذبه و وزن هواپیما است. نیروهای سطحی به سبب عبور جسم در میان سیال بوجود می‌آید. یعنی بین هوا و سطوح هواپیما، نیروی بَرا، پَسا و پیشرانه که سه نیروی اصلی عمل کننده بر روی هواپیما هستند همان نیروهای سطحی محسوب می‌شوند و لذا ۴ نیروی اصلی عبارتند از وزن، پیشرانه، بَرا و پَسا.

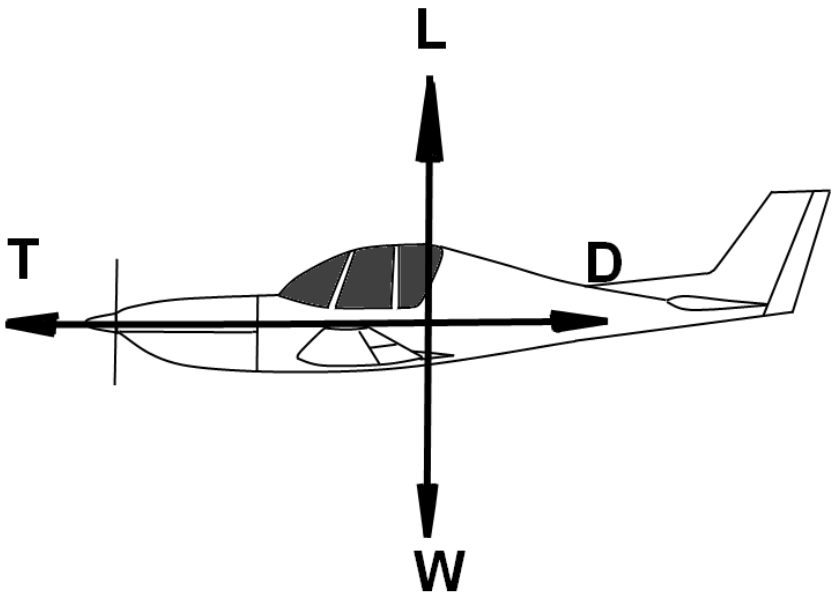
هر یک از نیروهای وارد بر هواپیما منشأ مشخصی دارند. نیروی محرکه توسط موتور مکانیکی تولید می‌شود و نیروی وزن ناشی از خاصیت جاذبه‌ی زمین است، اما منشأ نیروهای آیرودینامیکی پیکربندی هواپیماست که در این بخش به بحث در مورد آنها خواهیم پرداخت. منشأ اصلی تولید نیروهای آیرودینامیکی بال هواپیماست و لذا در محاسبات، نقش مهمی دارد. نقطه اثر نیروی آیرودینامیکی مرکز فشار cp نامیده می‌شود. به دلیل سهولت استفاده در محاسبات، نقطه اثر این نیرو را از مرکز فشار به نقطه‌ی دیگری منتقل می‌کنند که مرکز آیرودینامیکی ac نامیده می‌شود. با انتقال نیرو از مرکز فشار به مرکز آیرودینامیکی یک گشتاور نیز تولید می‌شود که این گشتاور در مقدار نیروی آیرودینامیکی تأثیری ندارد. از خصوصیات مهم این مرکز آن است که مقدار گشتاور تولید شده ناشی از انتقال نیرو به تغییر زاویه‌ی حمله بستگی ندارد. محل این مرکز اغلب در $\frac{1}{4}$ وتر از لبه‌ی حمله‌ی ایرفویل قرار دارد.

نیروی آیرودینامیکی معمولاً به دو مؤلفه تقسیم می‌شود. یک مؤلفه در راستای مسیر حرکت هواپیما و یک مؤلفه در راستای عمود بر مسیر حرکت هواپیما تعیین می‌گردد. شکل هندسی ایرفویل طوری است که همیشه یک مولفه به سمت بالا و یک مولفه در جهت مخالف مسیر حرکت قرار می‌گیرد.

به نیروی تصویر شده در راستای حرکت هواپیما، که معمولاً در خلاف جهت آن است، نیروی پسا D گویند. به نیروی تصویر شده در راستای عمود بر حرکت هواپیما نیز نیروی برا L گفته می‌شود. به طور کلی حدود ۹۰٪ تا ۱۱۰٪ نیروی برا توسط بال تولید می‌شود در حالی که فقط ۳۰٪ نیروی پسا توسط بال تولید می‌شود.

به طور کلی چهار نیروی اصلی بر هواپیمای موتوردار در حال پرواز وارد می‌شود که عبارتند از:

۱. نیروی وزن W
۲. نیروی جلوبرنده T
۳. نیروی آیرودینامیکی برا L
۴. نیروی آیرودینامیکی پسا D



شکل ۴ - چهار نیروی اصلی وارد بر هواپیما

دو نیروی آیرودینامیکی برا و پسا به عوامل زیر بستگی دارند:

۱. پیکربندی
۲. زاویه ی حمله
۳. ابعاد هواپیما
۴. سرعت حرکت

۵. چگالی هوا
۶. عدد رینولدز جریان هوا
۷. لزجت هوا

بنابراین رابطه‌ی برا و پسا را با عوامل ذکر شده می‌توان به شکل زیر در نظر گرفت:

$$L = \frac{1}{2}\,\rho\, V^2 S\, C_L,$$

$$D = \frac{1}{2}\,\rho\, V^2 S\, C_D.$$

در روابط فوق V سرعت بر حسب m/sec یا ft/sec، ρ چگالی هوا بر حسب kg/m^3 یا $Slug/ft^3$ و L و D به ترتیب نیروی برا و نیروی پسا بر حسب N یا lb هستند. S بیانگر مساحت ناخالص بال یعنی مجموع مساحت‌های بال در دو طرف بدنه و قسمتی از بدنه که حد فاصل بین بال‌هاست، و واحد آن m^2 یا ft^2 است. به این مساحت، مساحت مرجع S_{ref} می‌گویند. متغیرهای C_L و C_D ضرایب بدون بُعد هستند که با روش‌های تحلیلی و یا با استفاده از تونل باد بدست می‌آیند.

۳ – ۲ محاسبه نیروی پسا و ضریب آن

محاسبه‌ی نیروی پسا یکی از پیش نیازهای تحلیل عملکرد هر هواپیماست. مشکل‌ترین بخش محاسبه‌ی نیروی پسا، چگونگی احتساب نقش پیکربندی و اجزای هواپیماست. هر چند هواپیما یک جسم سه بعدی است ولی به دلیل رعایت واحدها در روابط ریاضی، به ناچار آن را با متغیرهای دو بعدی نمایش می‌دهیم.

ضریب پسا هرچند بی‌بُعد است ولی در درونش تأثیرات ابعاد هواپیما منظور شده است. همانطور که در این فصل خواهید دید، این ضریب مجموع دو متغیر است که یکی از آنها ضریب پسای برا صفر C_{D_0} نام دارد. زمان‌برترین و حساس‌ترین قسمت محاسبه‌ی نیروی پسا، محاسبه‌ی این ضریب است.

روش‌های مختلفی برای محاسبه‌ی ضریب C_{D_0} وجود دارد. اغلب شرکت‌های بزرگ سازنده‌ی هواپیما برای خود روش ویژه‌ای دارند که مبتنی بر روش‌های تجربی است. محاسبه‌ی دقیق این ضریب در پرواز آزمایشی امکان‌پذیر است ولی قبل از ساخت هواپیما، طراح به محاسبه‌ی ضریب C_{D_0} طرح خود نیاز دارد، لذا به کارگیری روش‌های تحلیلی ضروری است. در این پروژه با یکی از روش‌های نسبتاً معتبر به محاسبه‌ی این ضریب می‌پردازیم.

۳ – ۳ انواع نیروی پسا

نیروی پسا با مجموع نیروهای مختلفی برابر است که می‌توان آنها را در دو بخش کلی تقسیم بندی کرد:

۱. نیروی پسای وابسته به نیروی برا یا پسای القایی (D_i)
۲. نیروی پسای برا صفر (D_0)

لذا می‌توان نوشت:

$$D = D_0 + D_i.$$

۳ – ۴ پسای وابسته به برا

همانطور که از نامش پیداست به مقدار نیروی برا بستگی دارد. هر چه برا تغییر کند مقدار این نیرو هم تغییر خواهد کرد. نیروی برا نیز وابسته به زاویه‌ی حمله است؛ بنابر این تغییر در زاویه‌ی حمله به تغییر مقدار پسا نیز می‌انجامد.

پسای وابسته به برا خود به دو بخش تقسیم می‌شود. بخش اول پسای القایی است که در اثر گردابه‌ها در روی بال تولید می‌شود. بخش دوم پسای ناشی از خاصیت تراکم‌پذیری هواست که در سرعت‌های کم (زیر صوت) ناچیز است. پسای تراکم‌پذیری در سرعت‌های بالای 0.7 ماخ قابل توجه است و باید محاسبه شود.

۳ – ۵ پسای برا صفر

این نوع پسا ناشی از شکل هواپیما و نیروی اصطکاکی تولید شده در اثر تماس ملکول‌های هوا با سطح هواپیماست. تمام اجزای هواپیما از جمله بال، بدنه، دم، چرخ‌ها، میله‌های زیر بال و بطور کلی اجزایی که به گونه‌ای با جریان هوا تماس مستقیم دارند در تولید مقدار این پسا سهیم هستند. تمام اجزای اصلی هواپیما که هوا با آنها بر خورد دارد در ضریب C_{D0} مؤثرند. بطور کلی داریم:

$$C_{D0} = C_{D0f} + C_{D0w} + C_{D0vt} + C_{D0fu} + C_{D0lg} + C_{D0e} + \cdots.$$

یعنی C_{D0} هواپیما برابر با مجموع C_{D0} اجزای اصلی هواپیماست. مقدار C_{D0} هر یک از اجزا مثبت است و لذا هر یک بخشی از C_{D0} کل هواپیما را تولید می‌کنند. در رابطه‌ی بالا متغیرهای C_{D0f}، C_{D0w}، C_{D0vt}، C_{D0ht}، C_{D0lg}، C_{D0e}، به ترتیب ضریب پسای برا صفر بدنه، بال، دم عمودی، دم افقی، ارابه فرود و موتور هستند. حال به محاسبه‌ی C_{D0} هر یک از اجزای هواپیما می‌پردازیم:

۳ – ۶ بدنه ی هواپیما

ضریب پسای بدنه‌ی هواپیما از رابطه‌ی زیر بدست می‌آید:

$$C_{D0f} = C_f \, f_{LD} \, f_M \, \frac{S_{wet_f}}{S},$$

در این رابطه، متغییر C_f ضریب اصطکاک پوسته نام دارد و عددی بی‌بُعد است. این ضریب بر طبق رابطه‌ی پرانتل–شلختینگ برای لایه‌های مرزی مغشوش و آرام به صورت زیر به دست می‌آید:

لایه مرزی مغشوش:

$$C_f = \frac{0.455}{(log_{10}(R))^{2.58}},$$

لایه مرزی آرام:

$$C_f = \frac{1.327}{\sqrt{R}}.$$

در هواپیماها، معمولاً لایه مرزی ترکیبی از آرام و مغشوش است ولی بیشترین مقدار لایه‌ی مغشوش است. برای اطمینان از محاسبات، می توان کل لایه‌ی مرزی جریان را مغشوش یا متلاطم فرض کرد.

همچنین در این رابطه R عدد رینولدز است. عدد رینولدز عددی بی‌بُعد است که در محاسبات مربوط به طراحی، تطبیق اعداد به دست آمده از تونل باد برای هواپیمای واقعی و نیز محاسبه‌ی پسا نقش مهمی دارد. مقدار پسا تابع عدد رینولدز است و با تغییرات عدد رینولدز پساهای مختلفی را خواهیم داشت. رابطه‌ی مربوط به عدد رینولدز به شکل زیر است:

$$R = \frac{\rho \, V \, L}{\mu} = \frac{VL}{\nu},$$

که در آن:

$$\nu = \frac{\mu}{\rho}.$$

در روابط فوق، متغیر ρ چگالی، بر حسب $\frac{kg}{m^3}$، V سرعت واقعی هواپیما بر حسب $\frac{m}{sec}$، μ لزجت دینامیکی بر حسب $\frac{kg}{\sec m}$، ν لزجت سینماتیکی بر حسب $\frac{m}{\sec}$ و L طولی از شیء است که در جهت جریانی است که متلاطم است. در مورد بدنه L همان طول بدنه است.

متغییر f_{LD} تابعی از نسبت طول به بدنه است، و عددی بدون بُعد است. این متغیر از رابطه‌ی زیر بدست می‌آید:

$$f_{LD} = \frac{\frac{60}{(L/D)^3} + 0.0025\,(\frac{L}{D})}{B}$$

در این رابطه L طول هواپیماست، D قطر بدنه و B متغیری است که مقدار آن در سرعت‌های زیر صوت، حدود و مافق صوت به ترتیب از روابط زیر بدست می‌آید:

$$B = \sqrt{1 - M^2} \qquad (0 < M <= 0.9)$$
$$B = 0.44 \qquad (0.9 < M <= 1.1)$$
$$B = 1 \qquad (0 < M <= 0.9).$$

در این روابط M سرعت هواپیما بر حسب عدد ماخ است. متغیر f_M که تابع عدد ماخ نامیده می‌شود، از رابطه‌ی زیر محاسبه می‌شود:

$$f_M = 1 - 0.08\,M^{1.45}.$$

همچنین متغیر S_{wetf} سطح جانبی یا خیس شده‌ی بدنه و S مساحت ناخالص بال است. مساحت ناخالص بال مجموع مساحت خالص بال و مساحت آن قسمت از بدنه است که در بین دو نیمه بال قرار دارد.

۳ – ۷ بال ، دم افقی و دم عمودی

ضریب پسای برا صفر بال و دم‌ها به گونه‌ای مشابه و با یک روش محاسبه می‌شود. زیرا دم افقی و دم عمودی همان بال با ابعاد کوچکتر است. بنابر این ضریب پسای برا صفر بال w، دم افقی t و دم عمودی vt به ترتیب از روابط زیر به دست می‌آید:

$$C_{D0_w} = C_{f_w} f_{tc_w} f_M \left(\frac{S_{wet_w}}{S}\right) \left(\frac{C_{d_{min_w}}}{0.004}\right)^{0.4},$$

$$C_{D0_t} = C_{f_t} f_{tc_t} f_M \left(\frac{S_{wet_t}}{S}\right) \left(\frac{C_{d_{min_t}}}{0.004}\right)^{0.4},$$

$$C_{D0_{vt}} = C_{f_{vt}} f_{tc_{vt}} f_M \left(\frac{S_{wet_{vt}}}{S}\right) \left(\frac{C_{d_{min_{vt}}}}{0.004}\right)^{0.4}.$$

در روابط فوق متغییر های C_f و f_M به ترتیب ضریب اصطکاک و تابع عدد ماخ است که از روابط ذکر شده بدست می‌آیند. اندیس‌های w، t و vt به ترتیب نشانگر بال، دم افقی و دم عمودی است. دقت شود که در محاسبه‌ی عدد رینولدز، وتر متوسط آیرودینامیکی بال یا دم (MAC) بجای متغیر L قرار داده شود؛ به عبارت دیگر برای محاسبه عدد رینولدز بال و دم به صورت زیر در می‌آید:

$$R = \frac{\rho V \bar{C}}{\mu} = \frac{V \bar{C}}{\nu},$$

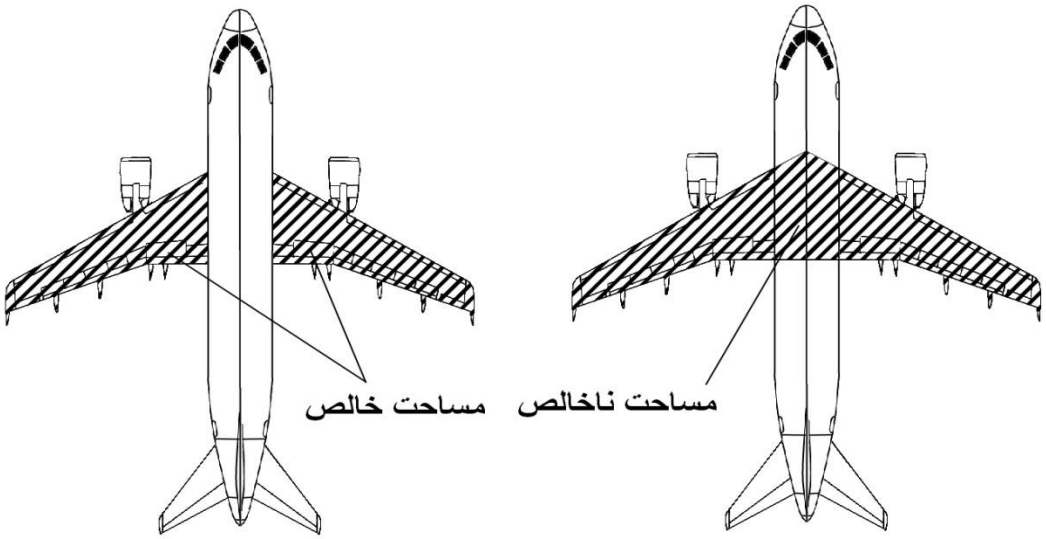
$$\bar{C} = MAC = \frac{2}{3} c_r \left(1 + \lambda - \frac{\lambda}{1 + \lambda}\right),$$

در این معادله c_r وتر ریشه و λ نسبت باریکی بال یا دم است.

متغیر f_{tc} تابعی از ضخامت بال یا دم است و از رابطه‌ی زیر بدست می‌آید:

$$f_{tc} = 1 + \frac{2.7 \left(\frac{t}{c}\right) + 100 \left(\frac{t}{c}\right)^4}{B},$$

در این رابطه t حداکثر ضخامت بال یا دم در محل وتر متوسط و C وتر متوسط MAC بال یا دم است. معمولا مقدار $\frac{t}{c}$ بال در حدود ۰/۱۲ تا ۰/۱۸ و برای دم در حدود ۰/۰۹ تا ۰/۱۲ است. متغییر S_{wet} را می‌توان حدود ۲ برابر مساحت بالای خالص بال فرض کرد . ضریب $C_{d\,min}$ حداقل مقدار C_d ایرفویل بال یا دم است.



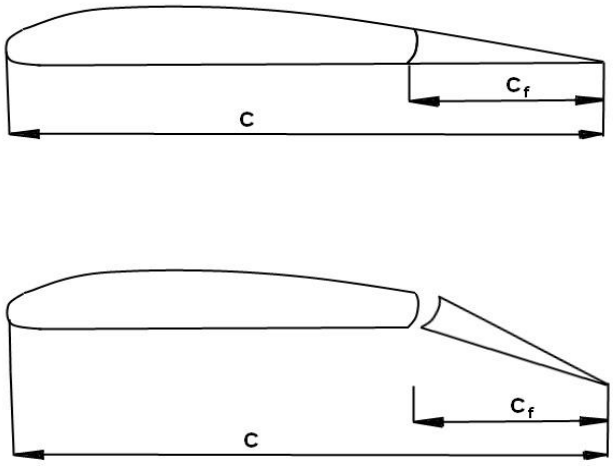
شکل ۵ – مساحت خالص و ناخالص بال

۳ – ۸ برافزا

برافزاها سطوحی هستند که برای افزایش قوس متوسط بال و نهایتاً به دست آوردن ضریب برای بیشتر استفاده می‌شوند. برافزاها بیشتر در حالات پروازی که سرعت کم است مانند فرود و برخاست به کار می‌روند. برافزا از نظر کاربرد دو نوع کلی است:

۱. برافزای لبه ی حمله LEHLD که عمدتاً به دو صورت شکاف‌دار و زائده در لبه‌ی حمله‌ی بال وجود دارند.
۲. برافزای لبه‌ی فرار TEHLD که فلپ نامیده می‌شوند و در لبه‌ی انتهایی بال واقعند.

از نظر طریقه‌ی عملکرد برافزاها می‌توان گفت که به عنوان بخشی از بال عمل کرده و در مواقع لزوم با استفاده از نیروی مکانیکی یا هیدرولیکی به سمت پایین چرخانده، و وارد عمل می‌شوند. برافزاها متناسب با نوع و وضعیت استفاده از آنها در حالت‌های برخاستن و فرود مقداری C_{D0} تولید می‌کنند.



شکل ۶ – وتر بال و برافزا در هنگام باز و پایین رفته

۳ – ۹ برافزای لبه‌ی فرار

ضریب پسای برا صفر این نوع برافزا در صورت پایین رفتن و متناسب با زاویه‌ی چرخش از رابطه‌ی زیر محاسبه می‌شود:

$$C_{D0_{flap}} = \left(\frac{C_f}{C}\right) A \left(\delta_f\right)^B.$$

در این رابطه کسر C_f/C نسبت وتر متوسط برافزا در حالت باز شده و پایین رفته به وتر متوسط بال است و معمولاً مقدار این کسر حدود ۰/۲ است. همچنین متغییر δ_f زاویه‌ی چرخش برافزا بر حسب درجه و A و B ضرایبی هستند که بسته به نوع برا افزا از جدول زیر بدست می‌آیند:

ردیف	نوع برافزا ی لبه ی فرار	A	B
۱	شکسته	۰٫۰۰۱۴	۱٫۵
۲	ساده	۰٫۰۰۱۶	۱٫۵
۳	تک شکافه (شکافدار)	۰٫۰۰۰۱۸	۲
۴	دو شکافه	۰٫۰۰۱۱	۱
۵	فولر	۰٫۰۰۰۱۵	۱٫۵

جدول ۱ – مقادیر ضرایب A و B انواع برافزای لبه‌ی فرار

۳ – ۱۰ برافزای لبه‌ی حمله

برای محاسبه‌ی ضریب پسای برا صفر، برافزای لبه‌ی حمله از رابطه‌ی زیر استفاده می‌کنیم:

$$C_{D0_{sl}} = \left(\frac{C_{sl}}{C}\right) C_{D0_w}.$$

نسبت $\left(\frac{C_{sl}}{C}\right)$ برابر نسبت وتر متوسط برافزای لبه‌ی حمله در حالت باز شده C_{sl}، به وتر متوسط بال C و نیز متغیر $C_{D0\,w}$ ضریب پسای برا صفر بال بدون استفاده از برافزاست.

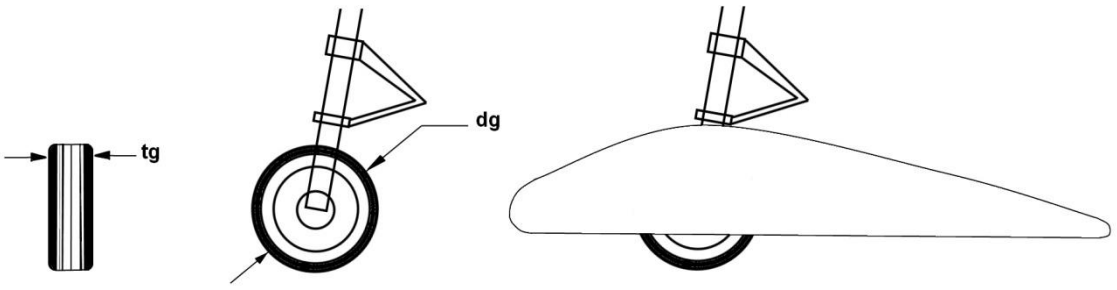
۳ – ۱۱ ارابه ی فرود

ارابه‌ی فرود یا چرخ‌ها در صورتی که هنگام برخاستن و فرود جمع نشده و یا از نوع ثابت باشند، در مقدار ضریب پسای برا صفر هواپیما تأثیر می‌گذارند و آن را افزایش می‌دهند. می‌توان اندازه‌ی ضریب پسای برا صفر ناشی از چرخ‌ها را از رابطه‌ی زیر بدست آورد:

$$C_{D0_{lg}} = \sum_{i=1}^{n} \left(C_{D_{lgi}} \frac{S_{lgi}}{S}\right).$$

در این رابطه S مساحت ناخالص بال، S_{lg} مساحت نمای جلوی هر چرخ که بردارهای جریان آزاد هوا به صورت قائم به آن برخورد می‌کند و $C_{D\,lg}$ ضریب پسای برا صفر هر چرخ است که این مقدار برای چرخ‌های بدون پوشش آیرودینامیکی ۰٫۳ و برای چرخ‌های پوشش دار ۰٫۱۵ می‌باشد. مساحت نمای جلویی هر چرخ برابر است با حاصلضرب قطر هر چرخ در ضخامت آن، و از رابطه‌ی زیر محاسبه می‌شود:

$$S_{tg} = d_g\, t_g.$$



شکل ۷ – ارابه فرود و پوشش آیرودینامیکی

۳ – ۱۲ میله ها یا ستونهای تقویتی بال

در بعضی هواپیماها برای تقویت سازه‌ی بال و کاهش وزن بال از ستون‌هایی استفاده می‌شود که در تماس با جریان هوا قرار دارد و تولید پسا می‌کند. این ستون‌ها معمولاً به صورت اریب بین بال و بدنه قرار دارند و بخشی از نیروی وارده بر بال به آنها منتقل می‌شود. در کایت‌ها از این ستون‌ها یا لوله‌ها بسیار استفاده می‌شود. برای کاهش C_{D0} این لوله‌ها مقطع آنها را به صورت ایرفویل در می‌آورند. هواپیماهای سسنا ۱۷۲ و سسنا ۱۸۵ در زیر هر بال خود دو ستون تقویتی دارند. امروزه به دلیل پیشرفت تکنولوژی ساخت سازه و دستیابی به کارایی بالاتر، کمتر از این ستون‌ها استفاده می‌شود. برای محاسبه‌ی ضریب پسای برا صفر این ستون‌ها از رابطه‌ی زیر استفاده می‌کنیم. البته باید توجه داشت که این رابطه برای محاسبه‌ی ستون‌هایی که چرخ‌ها به آنها متصل است نیز قابل استفاده است:

$$C_{D0_s} = \sum_{i=1}^{n} C_{D0_{si}} \left(\frac{S_s}{S}\right),$$

در این معادله S_s مساحت روبروی هر میله یا ستون (حاصلضرب قطر میله در طول آن) S مساحت ناخالص بال و $C_{D0\ s}$ ضریب پسای برا صفر هر میله است. ضریب $C_{D0\ si}$ هر میله برای میله‌هایی که سطح مقطع‌شان غیر ایرفویلی است برابر با ۱ و برای میله‌هایی که سطح مقطع‌شان آیرودینامیکی است برابر ۰٫۱ است. n نیز تعداد ستون‌ها یا میله‌ها است. ملاحظه می‌شود ایرفویلی نمودن سطح مقطع میله‌ها باعث کاهش ضریب پسای برا صفر آنها می‌شود. به عبارت دیگر آیرودینامیکی کردن ستون‌ها و میله‌ها حدود ده برابر ضریب C_{D0} آنها را کاهش می‌دهد. هر چند رعایت اصول آیرودینامیک در ساخت میله‌هایی که در معرض هوا هستند سبب افزایش وزن هواپیما می‌شود، ولی از طرف دیگر سود کاهش پسا بسیار بیشتر از ضرر افزایش وزن است.

۳ – ۱۳ ضریب C_{D0} کل

محاسبه‌ی تأثیر عواملی مانند عوامل ذکر شده آسان نیست و می‌توان از منابع تخصصی‌تر کمک گرفت. از آنجا که مجموع این عوامل بطور تقریبی حدود ۱۰ تا ۴۰ درصد در افزایش مقدار کل پسا مؤثرند، می‌توان با ضرب کردن ضریب تصحیح K_c در مجموع کل ضریب پساهای برا صفر اجزای هواپیما که از طریق معادلات خاص خودشان محاسبه شده‌اند، تأثیر عوامل بالا را در مقدار ضریب پسای برا صفر کل هواپیما، لحاظ کرد.

ردیف	نام جزء هواپیما	C_{D0}	درصد از کل
۱	بال	۰/۰۰۵۳	۲۳/۴۵
۲	بدنه	۰/۰۰۶۳	۲۷/۸۸
۳	مخازن سوخت نوک بال	۰/۰۰۲۱	۹/۲۹
۴	پوسته موتور	۰/۰۰۱۲	۵/۳۱
۵	پایه اتصالی موتور به بدنه	۰/۰۰۰۳	۱/۳۳
۶	دم افقی	۰/۰۰۱۶	۷/۰۸
۷	دم عمودی	۰/۰۰۱۱	۴/۸۶
۸	دیگر اجزای متفرقه	۰/۰۰۴۶	۲۰/۳۶
۹	کل هواپیما	۰/۰۲۲۶	۱۰۰

جدول ۲ - مقدار C_{D0} اجزای اصلی هواپیمای Gates learjet 25

بنابر این برای محاسبه‌ی ضریب پسای برا صفر کل هواپیما خواهیم داشت:

$$C_{D0} = K_c \left(C_{D0_w} + C_{D0_f} + C_{D0_t} + \cdots \right).$$

ضریب تصحیح K_c بستگی به نوع هواپیما، شکل کلی هواپیما و جنس و صافی سطوح خارجی آن دارد. جدول زیر مقدار این ضریب را برای تعدادی هواپیما نشان می‌دهد.

ردیف	نوع هواپیما	ضریب تصحیح K_c
۱	مسافری	۱/۱
۲	کشاورزی	۱/۵
۳	باری	۱/۲
۴	یک موتوره پیستونی	۱/۳
۵	عمومی هوانوردی	۱/۲ تا ۱/۴
۶	جنگنده	۱/۱

جدول ۳ - مقدار ضریب تصحیح K_c انواع هواپیما

فصل چهارم: شبیه‌سازی

۴ – ۱ شبیه‌سازیِ Real Time

در این بخش به معرفی شبیه سازی می‌پردازیم. از آنجایی که این مباحث بسیار پیشرفته هستند و نیاز به ریاضیات پیشرفته دارند، یک روش ساده انتخاب شده، تا به صورت Real Time در فضای ۳ بعدی شبیه‌سازی انجام گیرد.

در اینجا شبیه‌سازی بلادرنگ به این معناست که حالت یک شیء یا اشیاء در حال پرواز را با استفاده از معادلات ذکر شده در فصل قبل محاسبه کرده و بدست آوریم. برای شبیه‌سازی از scriptهای از قبل نوشته شده برای حرکتِ جسم استفاده نمی‌شود؛ در عوض از مدل فیزیکی ارائه شده و معادلات حرکت، و حل کننده معادلات دیفرانسیل برای پیشبردن شبیه‌سازی استفاده می‌شود.

این نوع شبیه‌سازی برای مدل‌سازی حرکت جسم صلب مانند هواپیما استفاده می‌شود. شاید یکی از مهم‌ترین و بنیادی‌ترین جنبه‌های پیاده‌سازی شبیه‌سازی بلادرنگ اجسام صلب، حل معادلات حرکت با استفاده از یک روش عددی انتگرال‌گیری مناسب باشد. به همین دلیل در ادامه‌ی این فصل به بررسی این موضوع می‌پردازیم.

۴ – ۲ انتگرال‌گیری از معادلات حرکت

معادلات دیفرانسیل حرکتی که تا بحال در مورد آنها بحث کردیم را می‌توان با انتگرال‌گیری حل کرد، و از آنها توابعی برای شتاب، سرعت، و مکان بدست آورد. ولی همان طور که در فصول قبل مشاهده کردید، محاسبات نیرو و گشتاور می‌تواند برای سیستم ما بسیار پیچیده باشد و یا از روی جداول تجربی بدست آید، که می‌تواند نوشتن برنامه‌ای ساده برای محاسبه‌ی انتگرال معین را دچار مشکل کند. به همین دلیل مجبوریم از روش‌های عددی برای تخمین زدن انتگرال معادلات حرکت بهره بریم. از واژه‌ی «*تخمین*» به این دلیل استفاده شده که بر پایه‌ی روش عددی انتخاب شده، جواب‌های بدست آمده دقیقاً با مقدار واقعی برابر نیستند و اندکی خطا خواهند داشت.

در اینجا یک توضیح کلی و غیر رسمی در مورد روش عددی بکار رفته، داده می‌شود، زیرا درک آن آسان‌تر است. به معادله‌ی دیفرانسیل حرکت خطی زیر نگاه کنید:

$$F = \frac{m\,dv}{dt}.$$

در این مثال ساده می‌توان این معادله را باز نویسی کرده و به صورت معین انتگرال‌گیری کرد:

$$\frac{dv}{dt} = \frac{F}{m},$$

$$dv = (F/m)\,dt.$$

می‌توان این معادله را به این صورت تعبیر کرد که تغییرات بسیار بسیار کوچک در سرعت dv، برابر است با (F/m) ضرب در تغییرات بسیار بسیار کوچک در زمان dt. در این مثال می‌توان با انتگرال‌گیری معین از طرف چپ معادله نسبت به سرعت و طرف راست نسبت به زمان، آن را حل کرد. روش عددی به کار گرفته شده این است که با فرض کردن یک مقدار بسیار بسیار کوچک تغییرات زمانی Δt می‌توان تغییرات بسیار بسیار کوچک سرعتΔv را بدست آورد:

$$\Delta v = (F/m)\,\Delta t.$$

نکته‌ی مهم این است که این روش به طور دقیق میزان سرعت را محاسبه نمی‌کند بلکه فقط یک تخمین از تغییرات سرعت را مشخص می‌کند. بنابر این برای محاسبه‌ی تقریبی سرعت واقعی جسم باید سرعت آن را قبل از تغییرات Δt بدانیم. در آغاز شبیه‌سازی و در زمان صفر باید مقدار سرعت اولیه‌ی جسم را مشخص کنیم. این سرعت اولیه به عنوان شرایط اولیه‌ی مورد نیاز برای هنگامی که زمان را بر اساس گام‌های Δt به پیش می بریم برای مشخص کردن سرعت جسم، با استفاده از معادله‌ی زیر به کار می‌رود:

$$v_{t+\Delta t} = v_t + (F/m)\,\Delta t,$$

که شرایط اولیه‌ی آن به صورت زیر است:

$$v_{t=0} = v_0.$$

در اینجا v_t برابر سرعت جسم در زمان t، $v_{t+\Delta t}$ برابر سرعت در زمان t به علاوه‌ی گام زمانی Δt، و v_0 سرعت اولیه در زمان صفر است.

از معادلات حرکت خطی می‌توان یکبار دیگر انتگرال‌گیری کرد تا معادله‌ی مکان جسم را نیز به طور تقریبی بدست آورد:

$$s_{t+\Delta t} = s_t + \Delta t\,(v_{t+\Delta t}),$$

که در آن شرایط اولیه مکان برابر:

$$s_{t=0} = s_0.$$

روش انتگرال‌گیری بحث شده‌ی اخیر به نام روش اویلر شهرت دارد، و یک روش پایه برای بیشتر انتگرال‌گیری هاست. این روش برای فهمیدن و درک کردن بسیار آسان است و از نظر پیاده‌سازی هم ساده می‌باشد، ولی لزوماً دقیق‌ترین روش به حساب نمی‌آید.

می‌توان به روشنی دریافت که هر چه میزان گام‌های زمانی Δt را کوچک‌تر و نزدیک‌تر به هم انتخاب کنیم، به جواب‌های دقیق‌تری خواهیم رسید. اما با انتخاب گام زمانی بسیار کوچک مشکلات محاسباتی بوجود خواهند آمد. بویژه نیاز به محاسبات بیشتری برای Δtهای کوچک نیاز خواهد بود، که در نهایت به دلیل گرد شدن یا از بین رفتن قسمت اعشاری، به یک خطای گرد شدن خواهیم رسید. این بدان معناست که در عمل با یک محدودیت برای اختیار کردن مقدارِ دقتِ گامِ زمانی مواجه هستیم. اما خوشبختانه روش‌های عددی مختلفی برای بالا بردن دقت انتگرال‌گیری نیز وجود دارند که می‌توان از آنها بهره برد.

۴ – ۳ روش اویلر

توضیحات قبل شرحی غیر رسمی از روش اویلر بودند. برای بررسی دقیقتر و ریاضیاتی‌تر این روش خوب است تا سری تیلور را برای بسط یک تابع کلی $y(x)$ بررسی کنیم. قضیه‌ی تیلور بیان می‌کند که با دانستن اطلاعاتی در مورد تابع و مشتق آن در برخی نقاط می‌توان مقدار تابع را تخمین زد. این تخمین با یک سری چند جمله‌ای نامتناهی بیان می‌شود:

$$y(x + \Delta x) = y(x) + (\Delta x)y'(x) + [(\Delta x)^2/2!]y''(x) + [(\Delta x)^3/3!]y'''(x) + \cdots,$$

که در آن y یک تابع مفروض بر حسب x است، $(x + \Delta x)$ مقدار جدید x است که می‌خواهیم تخمین بزنیم، y' مشتق مرتبه ی اولِ y ، y'' مشتق مرتبه‌ی دومِ y، y''' مشتق مرتبه ی سومِ y و به همین ترتیب می‌باشند. حال بجای $y(x)$ می‌توان $v(t)$ را جایگزین کنیم، که نتیجه می‌دهد:

$$v(t + \Delta t) = v(t) + (\Delta t)v'(t) + [(\Delta t)^2/2!]v''(t) + [(\Delta t)^3/3!]v'''(t) + \cdots.$$

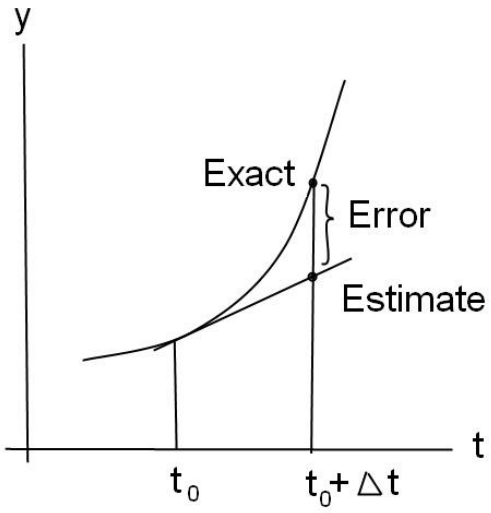
توجه داشته باشید که $v'(t)$ برابر dv/dt، که برابر F/m در معادلات حرکت می‌باشد. همچنین توجه کنید که مقدار v را در زمان t می‌دانیم . آنچه که می‌خواهیم بیابیم مقدار v در زمان $t + \Delta t$

است. با دانستن v در زمان t و مشتق آن در زمان t به عنوان یک تخمین اولیه. از آنجایی که مقدار مرتبه‌ی دوم و سوم و بالاتر را نمی‌دانیم، می‌توانیم آن قسمت‌ها را از چند جمله‌ای حذف کنیم. در نتیجه خواهیم داشت:

$$v(t + \Delta t) = v(t) + (\Delta t)v'(t).$$

این فرمول انتگرال اویلر، همان است که در قسمت قبل دیدیم. بخاطر اینکه فقط شامل مشتق مرتبه‌ی اول است، بقیه‌ی جملات سری که حذف شده‌اند برابر با میزان خطای محاسباتیِ truncation error محسوب می‌شوند. این قسمت‌هایی که حذف شدند عبارات مرتبه‌ی بالاتر خوانده می‌شوند و با حذف آنها نتیجه‌ی حاصل، تخمین مرتبه‌ی اول نامیده می‌شود. منطق نهفته در این تخمین این است که با جلو رفتن در سری، عبارات کوچکتری حاصل می‌شود که تأثیر کمتری در نتیجه نهایی دارند. به خاطر اینکه فرض کردیم Δt مقدار بسیار بسیار کوچکی دارد، Δt^2 از آنهم کوچکتر می شود، Δt^3 حتی از آنها هم مقدارش کوچکتر می‌شود، و به همین ترتیب، و بخاطر اینکه این عبارات Δt در صورت کسر قرار دارند، هر عبارت متوالی از مراتب بالاتر کوچکتر و کوچکتر می‌شوند. بنابر این مرتبه‌ی خطای این روش از مرتبه‌ی دوم $(\Delta t)^2$ خواهد بود زیرا اولین عبارت حذف شده‌ی غالب در این عبارت $[(\Delta t^2/2!]\, v''(t)$ از مرتبه‌ی دوم است.

از نظر هندسی، روش اویلر مقدار جدید را بر مبنای گام فعلی برای تابع مفروض بر اساس برون‌یابی مشتقات تابع در گام‌های قبلی محاسبه می‌کند. شکل زیر این روش را به تصویر می‌کشد:



شکل ۸ – گام انتگرال‌گیری اویلر

شکل بالا خطای حذف چندجمله‌ای‌های مراتب بالاتر را نشان می‌دهد. به وضوح مشخص است که با کاهش اندازه‌ی گام، اندازه‌ی قطعات چند جمله‌ای‌ها افزایش می‌یابد و تخمین بهتری از تابع حاصل می‌شود. همان طور که قبلاً گفته شد، این کار همیشه کارآمد نیست، زیرا مقدار محاسبات افزایش می‌یابد و خطای گرد کردن اعداد به سرعت انباشته می‌شود و در محاسبات انتشار می‌یابد.

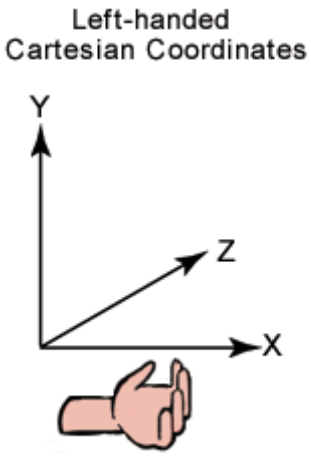
فصل پنجم: شبیه‌سازی هواپیما

۵ – ۱ شبیه‌سازی جسم صلب در فضای ۳ بعدی

در اینجا با شبیه‌سازی یک هواپیمای فرضی در فضای ۳ بعدی تمامی آنچه را که تا کنون بحث کردیم را به طور عملی می‌آزماییم و جمع بندی می‌کنیم. هواپیمای مفروض یک هواپیمای با پیکربندی معمولی به شکل هواپیمای سسنا ۱۷۲ می‌باشد. بال‌ها در قسمت جلو، بالابر یا دم افقی در پشت و یک دم عمودی در قسمت عقب و فلپ‌هایی در قسمت لبه‌ی فرار بال قرار دارد. کدهای ارائه شده، پیاده‌سازی مباحث شبیه‌سازی را شامل می‌شود. نمایش و رندر شدن مدل‌های سه بعدی توسط تکنولوژی Microsoft Direct3D انجام می‌شود.

مهم‌ترین بخش‌ها در این شبیه‌سازی شامل مدل هواپیما، انتگرال‌گیری، ورودی‌های کاربر، و نماش یا render کردن صحنه می‌باشند. باید توجه کرد که مدل ۳ بعدی هواپیما که برای نمایش از آن استفاده می‌شود تنها برای به تصویر کشیدن و ایجاد جلوه‌ی بصریِ نتیجه‌ی شبیه‌سازی به کار می‌رود، در حقیقت این انتگرال‌گیری است که **قلب شبیه‌سازی** را تشکیل می‌دهد و بسته به روش به کار رفته برای آن است که معادلات دیفرانسیل حرکت محاسبه می‌شوند و شبیه‌سازی در طول زمان به پیش می‌رود. در واقع ورودی‌های کاربر و نمایش ۳ بعدی عناصر و راه‌هایی هستند که به کاربر اجازه می‌دهند تا با شبیه‌ساز تعامل داشته باشد و شبیه‌سازی را ببیند. برای این منظور از keyboard برای دریافت ورودی از کاربر و از Direct3D برای نمایش استفاده می‌شود.

سیستم مختصات شبیه‌سازی همانند Direct3D به صورت چپگرد است و جهت محورهای آن به شکلی است که در تصویر زیر مشاهده می‌شود.



شکل ۹ - سیستم مختصات

در این سیستم جهت مثبت محور xها سمت راست، و جهت مثبت محور yها به سمت بالا است. برای مشخص کردن جهت مثبت محور zها از دست چپ کمک می‌گیریم، به این صورت که انگشتان را به سمت جهت محور xها باز کرده و جهت خم شدن آنها جهت مثبت محور yها را نشان می‌دهد. جهتی که انگشت شست به آن اشاره می‌کند جهت محور zها را نشان می‌دهد. چون این شبیه‌سازی در فضای ۳ بعدی انجام می‌شود با شروع شبیه‌سازی می‌توان هواپیما را به هر سمتی حرکت داد، می‌توان چرخید، صعود کرد یا به طرف پایین شیرجه زد یا هر نوع مانور دیگری را انجام داد.

۵ – ۲ مدل‌سازی و پیاده‌سازیِ جسم صلب

یکی از مهمترین جنبه‌های این شبیه‌سازی، مدل‌سازی پرواز است. بیشترین زمان این فاز از پروژه صرف تحقیق و بررسی و مدل‌سازی پرواز شده است. در مورد معادلات و نحوه‌ی محاسبه‌ی نیروها و گشتاورهای وارد بر یک هواپیما می‌توا نید به فصل‌های قبل مراجعه کنید.

برای پیاده‌سازی مدل پرواز، ابتدا باید یک structure برای کپسوله کردن تمام اطلاعات مورد نیاز برای مشخص کردن حالت یک جسم صلب در هر لحظه از زمان در طول شبیه‌سازی، آماده شود. برای این منظور یک struct با نام RigidBody به صورت زیر تعریف شده است:

جدول ۴ - ساختار RigidBody

```
struct RigidBody
{
    public float fMass;
    public float fMassInverse;
    public Matrix mInertia;
    public Matrix mInertiaInverse;
    public Vector3 vPosition;
    public Vector3 vVelocity;
    public Vector3 vVelocityBody;
    public Vector3 vAngularVelocity;
    public Vector3 vEulerAngles;
    public float fSpeed;
    public Quaternion qOrientation;
    public Vector3 vForces;
    public Vector3 vMoments;
    public Vector3 vAcceleration;
}
```

قدم بعدی برای پیاده‌سازی مدل پرواز آماده کردن تابعی برای مقدار اولیه دادن و initialize کردن هواپیما و پارامترهای مربوط به آن برای آغاز شبیه‌سازی است. برای این منظور تابع InitializeAirplane به شکل زیر نوشته شده است:

جدول ۵ - تابع InitializeAirplane

```
void InitializeAirplane()
{
    float iRoll, iPitch, iYaw;
    // Set initial position
    Airplane.vPosition = new Vector3(15000, 500, 5000);
    // Set initial velocity
    Airplane.vVelocity = new Vector3(0,0,100);
    Airplane.fSpeed = 100;
    // Set initial angular velocity
    Airplane.vAngularVelocity = new Vector3(0, 0, 0);
    // Set the initial thrust, forces, anfd moments
    Airplane.vForces = new Vector3(0, 0, 500);
    ThrustForce = 500;
    Airplane.vMoments = new Vector3(0, 0, 0);
    // Zero the velocity in body space coordinates
    Airplane.vVelocityBody = new Vector3(0, 0, 0);
```

```
        // Set these to false at first,
        //you can control later using the keyboard
        Stalling = false;
        Flaps = false;
        // Set the initial orientation
        iRoll = 0;
        iPitch = 0;
        iYaw = -3.14f/2.0f;
        Airplane.qOrientation =
            Quaternion.RotationYawPitchRoll(iYaw, iPitch, iRoll);
        Airplane.vVelocity.TransformCoordinate(
            Matrix.RotationQuaternion(Airplane.qOrientation));
        // Now go ahead and calculate the plane's mass properties
        CalcAirplaneMassProperties();
    }
```

این تابع مکان اولیه، سرعت، ارتفاع و نیروی پیشرانه را برای هواپیما مقدار دهی می‌کند و تابع CalcAirplaneMassProperties را برای محاسبه‌ی خواص مربوط به جرم جسم که همان گشتاور اینرسی هواپیماست را فراخوانی می‌کند. این محاسبات بر این اساس انجام می‌شوند که هواپیما برای مدل‌سازی به قسمت‌های مختلف سازه‌اش تقسیم می‌شود، برای مثال بال‌ها، سکان عمودی، دم افقی و بدنه‌ی هواپیما. و برای هر قسمت به طور مجزا گشتاور اینرسی آن محاسبه شده و در نهایت با هم ترکیب می‌شوند تا گشتاور اینرسی کل هواپیما محاسبه شود. تابع InitialAirplane در ابتدای برنامه هنگامی که پنجره‌ی برنامه اصلی ایجاد می‌شود در سازنده‌ی کلاس فراخوانی می‌شود.

آخرین بخش از شبیه‌سازی به مدل پرواز مربوط می‌شود به محاسبه‌ی نیرو ها و گشتاورهای وارد شده به هواپیما که در طول شبیه‌سازی در هر لحظه از زمان به آن وارد می‌شود. به این منظور تابع CalcAirplaneLoads در هر گام از شبیه‌سازی فراخوانی می‌شود. این تابع از توابع دیگری برای محاسبه‌ی ضرایب آیرودینامیکی بال، دم افقی و دم عمودی تشکیل شده است. ابتدا تمام نیروها در سیستم مختصات ثابت محلی هواپیما محاسبه می‌شوند و سپس قبل از اعمال نیروی جاذبه به سیستم مختصات زمین تبدیل می‌شوند.

۵ – ۳ انتگرال‌گیری

بعد از محاسبه‌ی نیروهای وارد بر هواپیما، نیاز داریم تا با انتگرال‌گیری از معادلات حرکت شبیه‌سازی را در طول زمان به پیش ببریم. اولین چیزی که باید در مورد آن تصمیم‌گیری شود این است که از چه روشی

برای محاسبات می‌خوا هیم استفاده کنیم. در اینجا با مباحث مطرح شده در فصل قبل، روش اویلر را به خاطر سادگی در محاسبات و پیاده‌سازی انتخاب می‌کنیم. تابع StepSimulation تمام محاسبات لازم برای انتگرال‌گیری را اداره می‌کند.

جدول ۶ - تابع StepSimulation

```
void StepSimulation(float dt,ref RigidBody Airplane)
{
    // Take care of translation first:
    // (If this body were a particle, this is all you would need to
    Vector3 Ae;
    // calculate all of the forces and moments on the airplane:
    CalcAirplaneLoads(ref Airplane);
    // calculate the acceleration of the airplane in earth space:
    Ae = Airplane.vForces * (1.0f / Airplane.fMass);
    Airplane.vAcceleration = Ae;
    // calculate the velocity of the plane in earth space
    Airplane.vVelocity += Ae * dt;
    // calculete the position  of the airplane in earth space:
    Airplane.vPosition += Airplane.vVelocity * dt;
    // Now handle the rotations:
    float mag;
    // calculate the angular velocity of the airplane in body space:
    Vector3 Iw = Airplane.vAngularVelocity;
    Iw.TransformCoordinate(Airplane.mInertia);
    Vector3 alpha = Vector3.Cross(Airplane.vAngularVelocity, Iw);
    alpha = Airplane.vMoments - alpha;
    alpha.TransformCoordinate(Airplane.mInertiaInverse);
    Airplane.vAngularVelocity += alpha * dt;
    // calculate the new rotation quaternion:
    Quaternion q =
        new Quaternion(
        Airplane.vAngularVelocity.X / 2 * dt,
        Airplane.vAngularVelocity.Y / 2 * dt,
        Airplane.vAngularVelocity.Z / 2 * dt,
        0);
```

```
    Airplane.qOrientation += q * Airplane.qOrientation;

    // now normalize the orientation quternion:
    mag = Airplane.qOrientation.Length();
    //if (mag != 0)
    //    Airplane.qOrientation /= mag;
    Airplane.qOrientation.Normalize();

    // calculate the velocity in body space:
    // (we'll need this to calculate lift and drag forces)
    Airplane.vVelocityBody = Airplane.vVelocity;
    Airplane.vVelocityBody.TransformCoordinate(
        Matrix.RotationQuaternion(
            Quaternion.Conjugate(Airplane.qOrientation)));
    // calculate the air speed:
    Airplane.fSpeed = Airplane.vVelocity.Length();
    // get the Eular angles for our information
    Vector3 u;
    u = MakeEulerAnglesFromQ(Airplane.qOrientation);
    Airplane.vEulerAngles = u;
}
```

اولین کاری که این تابع انجام می‌دهد تابع CalcAirplaneLoads را برای محاسبه‌ی بار و نیروهای وارد بر هواپیما در لحظه‌ی فعلی از زمان، فراخوانی می‌کند. سپس به محاسبه‌ی شتاب خطی وارد بر هواپیما بر اساس نیروهای وارده که اخیراً محاسبه شدند می‌پردازد. سپس به انتگرال‌گیری به روش اویلر می‌پردازد. ابتدا سرعت خطی هواپیما را محاسبه می‌کند و بعد از آن مکان آن را بدست می‌آورد. سپس محاسبات مربوط به حرکت دورانی را انجام می‌دهد و جهت هواپیما را مشخص می‌کند.

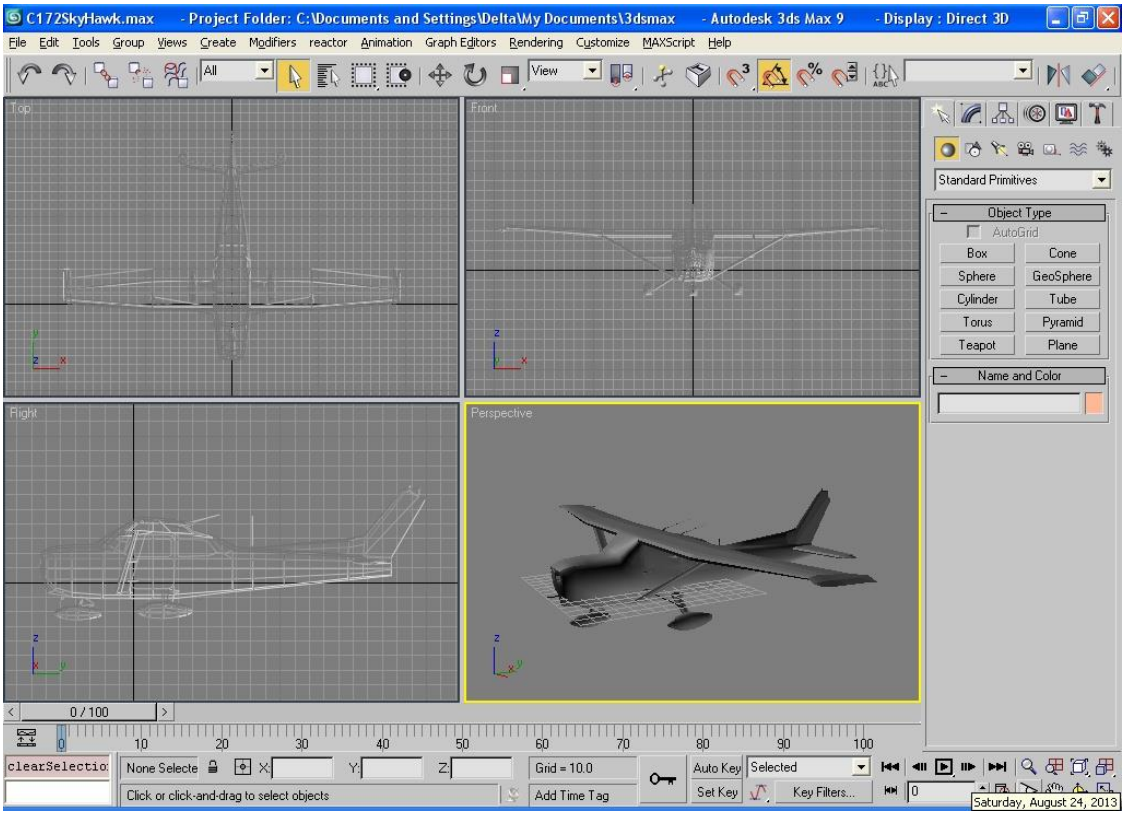
۵ – ۴ هدایت و کنترل هواپیما

تا این جا، شبیه‌سازی هنوز کامل نشده است؛ به خاطر اینکه هنوز برای کنترل هواپیما تدبیری اندیشیده نشده است. برای تعامل و کنترل هواپیما نیاز به پیاده‌سازی سازوکارهای کنترل هواپیما داریم. دستگاه ورودی اصلی به کار برده شده Keyboard می‌باشد. باید به یاد داشته باشیم که در شبیه‌سازهای برپایه‌ی فیزیک مانند این پروژه، نباید به طور مستقیم حرکت هواپیما را کنترل کرد؛ کنترل هواپیما تنها با اعمال نیروهای مختلفی که بر هواپیما اثر میگذارند، و با انتگرال‌گیری در طول زمان امکان پذیر است.

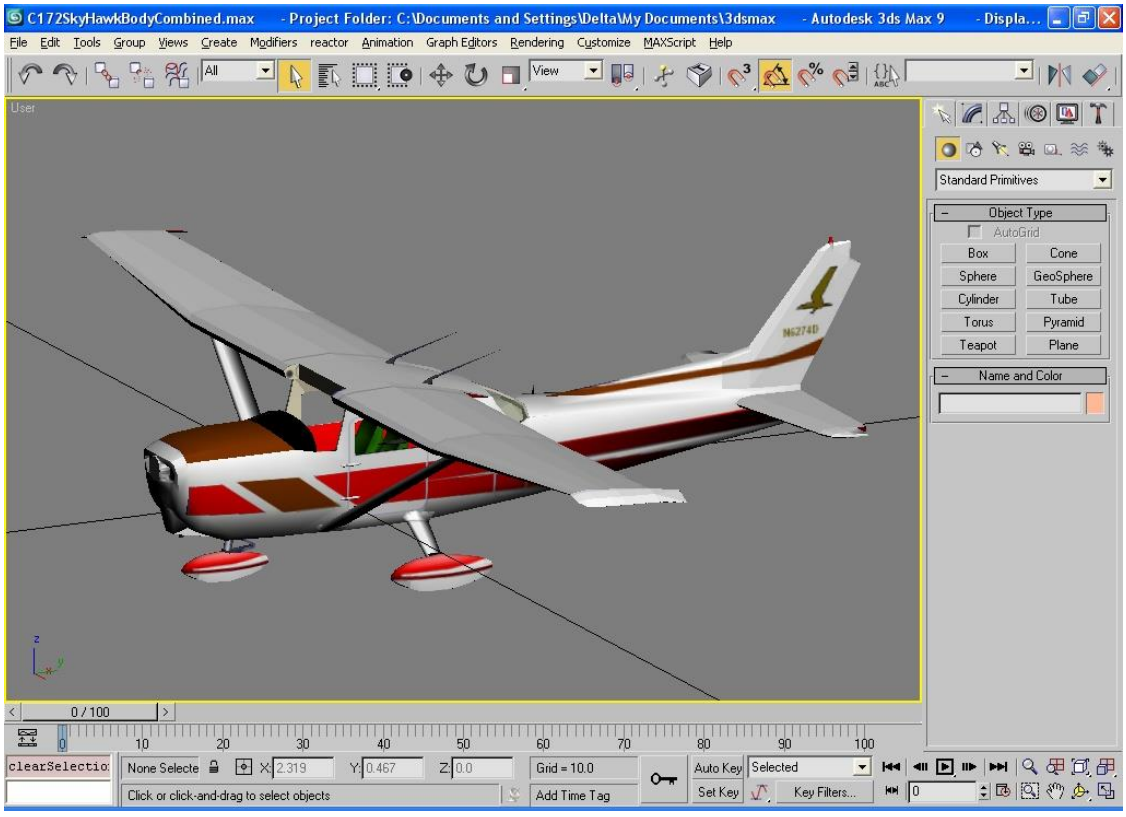
برای شبیه‌سازی stick از کلیدهای جهتی استفاده شده. با فشار دادن کلید پایین، دسته به سمت عقب کشیده می‌شود و دماغه‌ی هواپیما به طرف بالا می‌رود. با فشار دادن کلید بالا، دسته به جلو هل داده می‌شود و دماغه به طرف پایین شیرجه می‌رود. کلید سمت چپ، غلتش به چپ و کلید سمت راست غلتش به راست را کنترل می‌کنند. کلید X سکان عمودی را به طرف چپ می‌گرداند و باعث گردش به چپِ دماغه‌ی هواپیما می‌شود. کلید C سکان عمودی را به طرف راست می‌گرداند و باعث گردش دماغه‌ی هواپیما به سمت راست می‌شود. کلید + باعث افزایش نیروی محرکه‌ی موتور می‌شود و سرعت هواپیما را افزایش می‌دهد. کلید - نیز نیروی محرکه هواپیما را کاهش می‌دهد و باعث کاهش سرعت هواپیما می‌شود. کلید F باعث باز و بسته شدن Flapها می‌شود.

۵ – ۵ مدل‌سازی ۳ بعدی و Renderکردن شبیه‌سازی

برای مدل‌سازی ۳ بعدی هواپیما از نرم افزار 3ds Max بهره گرفته شده است. ابتدا مدل هواپیمای sesna 172 sky hawk بر اساس تصاویر سه نمایه‌ی مرجع ساخته شده و سپس بافت و texture به آن افزوده شده است. برای استفاده از آن در برنامه شبیه‌ساز ، یک converter برای تبدیل فایل‌های ذخیره شده به فرمت 3ds. که نرم افزار 3ds Max از آن برای ذخیره کردن مدل استفاده می‌کند نوشته شده. این برنامه‌ی تبدیل کننده، فایل باینری با فرمت 3ds. را به فرمت text تبدیل می‌کند و برنامه‌ی شبیه‌ساز در ابتدای شبیه‌سازی، مدل را بارگیری کرده و از آن برای نمایش نتیجه حاصل از شبیه‌سازی استفاده می‌کند. بدین ترتیب با استفاده از این برنامه می‌توان تمام مدل‌های سه بعدی ساخته شده در برنامه‌ی 3ds Max را با استفاده از parserی که در برنامه شبیه ساز نوشته شده بارگیری کرد و از آنها برای render کردن صحنه بهره گرفت. دو عدد دوربین در برنامه شبیه‌ساز در نظر گرفته شده که با کلید های F1 و F2 می‌توان به ترتیب بین دوربین‌های شماره‌ی ۱ و شماره‌ی ۲ جابجا شد. دوربین ها از نوع Chase هستند. دوربین شماره‌ی ۱ در پشت هواپیما قرار دارد و دوربین شماره‌ی ۲ می‌تواند آزادانه در حول هواپیما حرکت کند و از تمام زوایا آن را مشاهده کند. تصاویر زیر مراحل مدل‌سازی را نشان می‌دهد.



شکل ۱۰ – مدل‌سازی ۳ بعدی

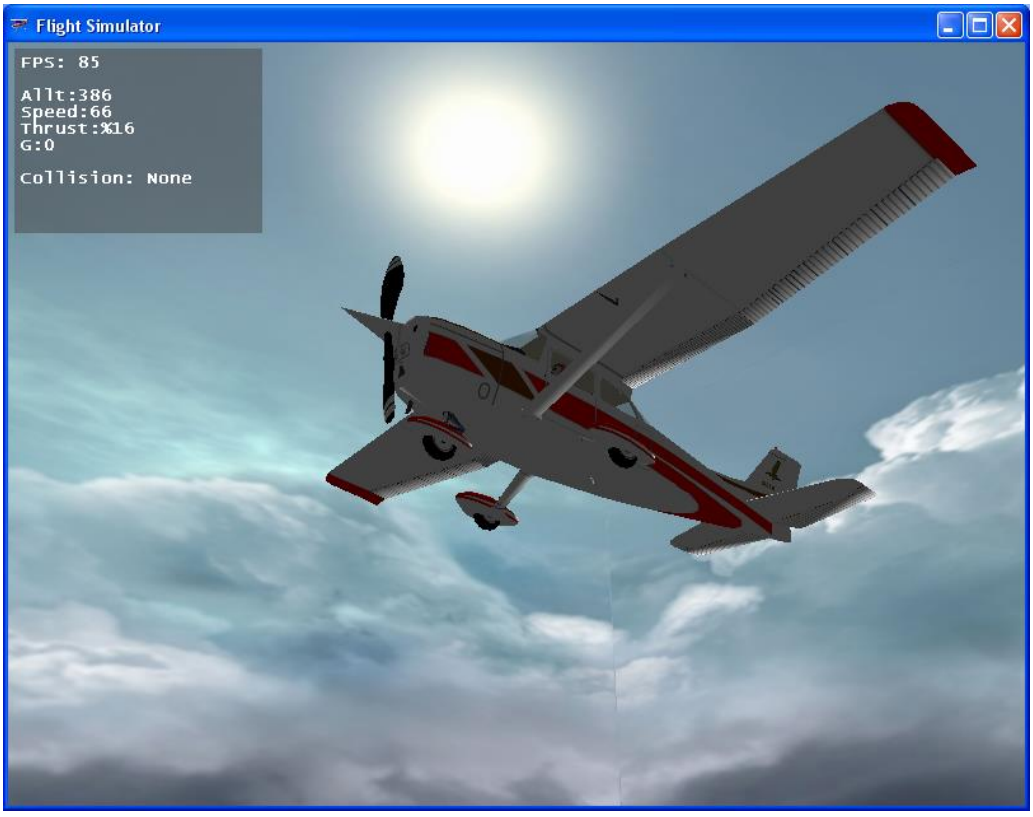


شکل ۱۱ – اختصاص texture به مدل

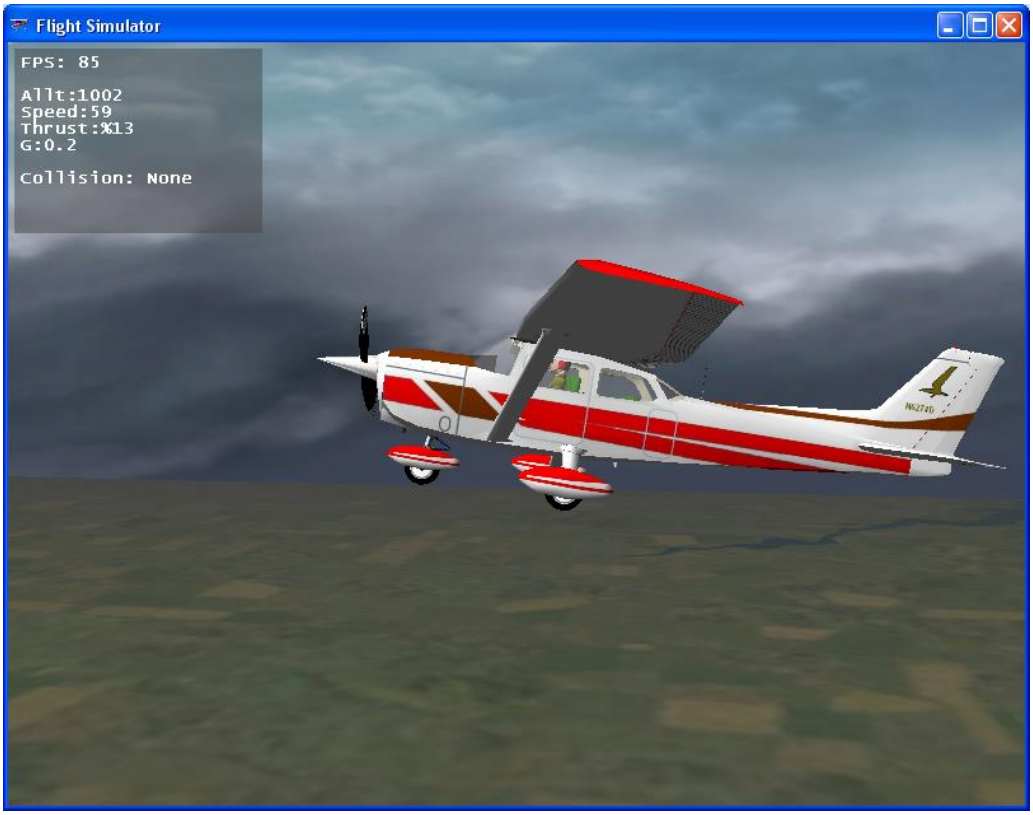
در نهایت با کنار هم قراردادن قطعات و مولفه‌های نوشته شده برای شبیه‌سازی و render کردن گرافیکی، برنامه‌ی شبیه‌ساز کامل شده و نتیجه‌ی حاصل از آن تا حد زیادی قابل قبول می‌باشد.

شکل ۱۲ - نمایی از برنامه‌ی شبیه‌ساز

شکل ۱۳ - نمایی از برنامه‌ی شبیه‌ساز



شکل ۱۴ - نمایی از برنامه‌ی شبیه‌ساز



شکل ۱۵ - نمایی از برنامه‌ی شبیه‌ساز

نتیجه‌گیری

در این پروژه با به کار گیری روش‌های عددی مناسب برای انتگرال‌گیری از معادلات دیفرانسیل حرکت و با تحقیق و مطالعه‌ی دقیقِ مباحث فیزیک، آیرودینامیک و مکانیک پرواز؛ و همچنین بهره گیری از بسته‌ی توسعه‌ی نرم افزارِ DirectX به مدل‌سازی و شبیه‌سازی پرواز یک هواپیمای فرضی پرداختیم. از مزایای استفاده از شبیه‌سازهای پرواز می‌توان به امنیت، مزایای اقتصادی و صرفه جویی در زمان اشاره کرد. علاوه بر استفاده از شبیه‌سازها در بخش‌های مختلف صنایع هوایی می‌توان از آنها در صنایع بازی‌سازی و Game نیز استفاده کرد. طبیعی است که پوشش دادن تمام فعالیت‌های انجام شده برای این نرم افزار در اینجا میسر نبوده؛ بلکه خلاصه‌ای از مهمترین و با اهمیت‌ترین فعالیت‌های صورت گرفته در راستای تکمیل این پروژه به طور اجمالی مطرح شده است. با توجه به اینکه نرم افزاری در این سطح و با در نظر گرفتن چشم انداز ذکر شده در بخش نخست این پایان‌نامه، نیاز به کار، فعالیت و تحقیقات فراوانی برای تکمیل نهایی آن هست. از این روی از افرادی که تمایل به همکاری در انجام و تکمیل هر بخشی از این نرم افزار اعم از بخش گرافیک، بخش برنامه نویسی، بخش تحقیقات و غیره دارند، دعوت به همکاری می‌شود. برای تماس با اینجانب می‌توانید با شماره‌ی ۰۹۳۸۴۳۰۶۸۶۷ و یا با رایانامه star_tehran_delta@yahoo.com تماس بگیرید و مرا از نظرات و پیشنهاداتان بی‌بهره نگذارید. اینجانب آمادگی دارم تا تمام تجربیاتی که کسب کردم و سورس کد برنامه‌هایی که برای تکمیل این پروژه نوشتم را در اختیار علاقمندان بگذارم.

منابع و ماخذ

فهرست منابع فارسی

۱. مهندس محمد هاشم صدرایی، مکانیک پرواز، دانشگاه امام حسین (ع) ، ۱۳۷۷.

۲. جان تئودور تالی، ترجمه دکتر محسن جهان میری، آیرودینامیک به زبان ساده (مصور)، انتشارات یا مهدی، چاپ دوم، ۱۳۸۷.

فهرست منابع لاتین

1. David M. Bourg, Physics for Game Developers, O'Reilly, First Edition, Jan 2002.
2. David Allerton, Principles of Flight Simulation, WILLEY, First Edition, Nov 2009.
3. Microsoft, DirectX documentation for managed languages (.NET), Feb 2007.
4. Microsoft, DirectX documentation for C++, Feb 2007.
5. Frank D. Luna, Intruduction to 3D Game Programming with DirectX 9.0, Wordware Publising Inc., 2003.