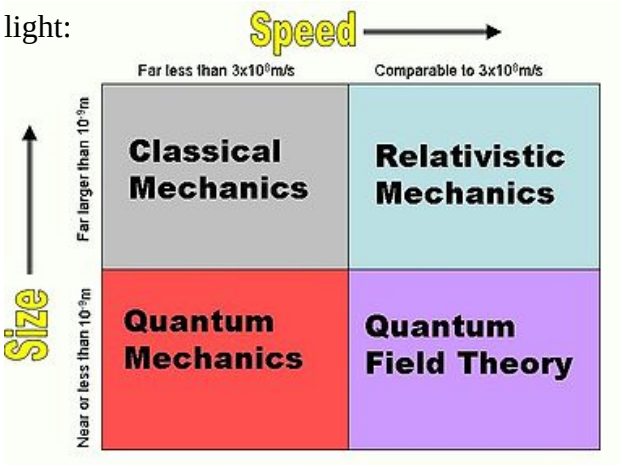
HET321 Physics of Games: Project

Research

Video game physics involves introducing the laws of physics into a game engine or simulation. Specifically Newtonian physics (The initial stage of classical mechanics), describing the motion of objects, either stationary or acted upon by a force. Quantum physics and relativity is most developed and most accurate forms of mechanics, however, these mainly involve objects smaller than 1 x 10 ^ -9 m, or moving close to the speed of light:



Classical mechanics: Wikipedia

Therefore quantum mechanics and relativity in not very appropriate for game physics, but Newtonian physics is more than appropriate:

First law:

"A body persists its state of rest or of uniform motion unless acted upon by an external unbalanced force." Often referred to as law of inertia.

Second law:

"Force equals mass times acceleration (**F** = *m***a**)". Where the net force of an object is calculated by the mass times acceleration. The force is proportional to the time rate of change of its linear momentum (**F** = d(*mv)/dt).*

Third law:

"To every action there is an equal and opposite reaction." As a force is applied to an object, an equal force with same magnitude and opposite direction is simultaneously exerted.

Velocity is described by the change of position with respect to change in time. $\vec{v} = \frac{\mathrm{d}\vec{r}}{\mathrm{d}t}$ Velocity however is given direction as well, and therefore it is a vector, unlike speed that without direction is a scalar.

Acceleration is the derivative of the velocity (rate of change of velocity) with respect to change in time. $\vec{a} = \frac{\mathrm{d}\vec{v}}{\mathrm{d}t}$

The net force on a particle is equal to the rate of change of momentum ($m\vec{v}$) with respect to change in time: $\vec{F} = \frac{\mathrm{d}\vec{p}}{\mathrm{d}t} = \frac{\mathrm{d}(m\vec{v})}{\mathrm{d}t}$. Newton simplified it as **F** = m**a.**

Potential energy(usually gravitational: $U = mgh$) is the energy that a particle has in order to gain kinetic energy(the energy of a moving body:$E_k = \frac{1}{2}mv^2$). Conservation of forces defines that total energy on a particle is the sum of the kinetic energy and potential energy.

Game engines are the core code within any game, simulating physics. Game engines generally try to mimic real life physics, through motion of objects/particles. But it is whether the games choice to use the entire engine or make it surreal to allow for fun gameplay. The dominating limitation to game/physics engines realism is the precise representation of an object in a space (exact locations are large numbers and are rounded). When rounding in engines is done, depending on the precision, object may overshoot or undershoot their target or current position. In addition to precision, framerate of a certain game/physics engine plays a defining roll, the number of moments in time per second when calculations are made. So when many calculations are made framerate is low and results in a non-smooth moving object which may look jagged and

even appear to be teleporting around. Therefore it is essential for game engines to be as optimized as possible, which in some cases means cutting corners in the physics calculations.

Collision detection is another important aspect of a Game engine; it involves algorithms for checking collisions between objects. Then simulates what happens when a collision is detected, ‘collision response’. Collision detection is done each frame, to ensure maximum accuracy, but because it involves calculations it is very taxing on CPU and as a result dropping framerate. Optimizing collision detection can occur in many ways, either by like game engines cutting corners and only checking ever few frames instead of each one or by only reducing the realism of physics involved.

Stunt Pilot (3rd person simulator): Review
-Flash game-

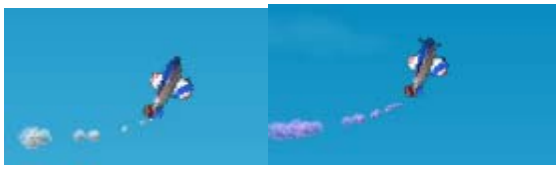
Stunt pilot is a 3rd person 2D side on plane simulator, where players take control of a plane and have to navigate through a small obstacle course of rings. The rings don't need to be completed in order, as long as the player goes through them all. There is a course time for each level, where players try to beat the clock in order to score more points. *Stunt Pilot* has an added function of a boost button that speeds up the plane, and while boosted gives bonus points to players while flying through the rings. The controls are very simple, up to turn the plane up and down to go down. However, it is hard to master due to the fact the plane can fly upside down and vertically, where up and down controls are in respect to the plane and not the screen.

Newton's laws of motion only play a small part of this game, due to the lack of controls over speed during gameplay. The plane starts with an initial thrust which stays constant; accelerating the plane to a velocity where the resistance forces (drag/friction) is equivalent to that of the thrust. This allows the plane to fly at a constant speed due to the net force of the plane equalling zero, and thus eliminating acceleration. However, when the planes direction is altered gravity acts as a force on the plane, either increasing or decreasing the net force applied to the plane, resulting in an acceleration or deceleration of speed. Again when the net forces equal that of the resistance forces the plane stays at a constant speed. This simulating real physics, however, while playing the game it doesn't fully feel real, due to either gravity, mass of the plane, the thrust or resistant forces not being values of which one would encounter in the real world, ie. The plane seams to reach maximum velocity too quickly.

The boost button increases the thrust exerted by the plane, increasing net force and altering velocity, where it accelerates to that velocity (simulating good physics behaviour), until the player stops pressing down boost at which point the plane returns to its original velocity through deceleration. Therefore we can say the plane is cruising, it

maintains a constant of two thrust forces, and only the direction affect the way the plane behaves.

Momentum is still imposed on the plane as it flies, this simulates smooth transition of motion as apposed to sharp, spontaneous, jagged change in velocity. It is clear that this is handled very well within *Stunt Pilot* due to the comparison in arcs when the plane is travelling at minimum speed and maximum speed:



The curve to which the plane changes direction is increased when speed is higher, this is because of change in momentum is lower when travelling at a higher speed (inertia or newtons first law).

We can assume mass is at a constant of 1, because there is no variety of planes or modifications to be done to the current plane or a fuel gauge (plane never runs out of fuel). This helps optimise the game engine due to all mass multiplications is done to a mass of 1, instead of a variable mass which would be more calculation hectic.

Stunt Pilot uses collision detection to tell if the plane has collided with an object. We assume it checks for collision each frame to maximise effectiveness. Once a collision has been detected the player loses control of the plane, which thrust is nullified and only the force of gravity takes over simulating a crash to the ground. The plane as an object does not get modified, unlike in real life.

As far as simulators go, *Stunt Pilot* is a naive example of the real world. It simulates smooth flowing motion by the act of forces, giving *Stunt Pilot* a good interpretation of what happens in a constant thrust object. However, due to the lack of thrust control and non realistic values, *Stunt Pilot* is a basic physics engine, which gives a good approximation to the effects of motion.

Bump Copter 2 (3rd person): review
-Flash game-

Bump Copter 2 is a 3rd person 2d side on helicopter simulator, where the player takes control of a helicopter and navigates through an obstacle course to reach a flag. The helicopter has a health status which gets damaged when the player collides with objects in the obstacle course. *Bump Copter* uses an interesting 'bump' physics, which when the helicopter collides with an object it is sent into the opposite direction with maximum force, pushing the helicopter away. The arrow keys control navigation for the helicopter, And the space bar is a shoot button, which fires a projectile in a straight line, at a constant speed, not being affected by force.

Gravity is completely neglected in *Bump Copter 2*; this becomes abundantly clear when the helicopter accelerates upwards just as fast as it does downwards with the same force. Mass of the helicopter is also neglect able, we assume a value of 1 is used because there is no variety of helicopters or modifications to be done to the current one or a fuel gauge (helicopter never runs out of fuel). This helps optimise the game engine due to all mass multiplications is done to a mass of 1, instead of a variable mass which would be more calculation hectic.

Bump Copter 2 uses Newton's laws of physics to the extent of making the game more fun than realistic. It uses the first law where the helicopter will stay at its current path (velocity) until either the play changes the force on it by the arrow keys or by a collision with an object. The arrow keys provide a certain constant force on the helicopter giving it a constant acceleration until the helicopter reaches its maximum speed at which it is clipped. However, forces are still applied on the helicopter, so when a different directional key is pressed the helicopter doesn't instantly change direction and velocity but rather arcs due to the resulting forces getting applied to the helicopter.

A 'bump' feature makes the game more dynamic, so instead of collision destroying the helicopter instantly or slowing it down, bumping into another object simply pushes the

helicopter away. The push is in the opposite direction in respect to the object, ie. If bumped into a vertical wall the helicopter is pushed away on the x axis, so the y axis forces stay the same. Collision between the outer walls and gameplay objects is different; when colliding with a game object the force exerted on the helicopter is opposite and equal, but when colliding with the outer walls, the force being applied is increased, so it's not an equal opposite force but a greater one. However, the helicopter can never exceed its maximum velocity and therefore it is usually clipped to maximum.

Bump Copter 2 is a poor example of a real physics engine, with the lack of gravity and improper collision response. Only good thing to take away from it is the smooth flow of motion of the helicopter and moving object within the game. Using proper force physics to increase acceleration and velocity.

Microsoft Flight Simulator X (first or third person): review
-PC-

*Flight Simulator X (*FSX) is the tenth evolution of the original Flight Simulator from Microsoft back in 1980. FSX is the closest simulator to real life flying of a plane on any platform; it uses an extensive physics engine to give players as real as possible experience. The player takes actual control of the cockpit of a plane, pressing all the buttons available to a real pilot of that plane, but at any time during gameplay players can change view of the plane to see the flight in action. FSX comes with a large variety of planes to fly, each with their own different cockpit control scheme and plane physics. This however, is often frustrating and overwhelming for new players due to its realism.

FSX is so real that it incorporates sophisticated weather simulations and varying air traffic environments. The forces acting on a plane varies depending on the weather, ranging from a thunder storm (providing extreme turbulence) to calm clear blue skies. This is taken into account when calculating velocity and acceleration, because the net forces acting on the plane are changing. Doing crazy stunts in the air that are not properly handled does result in the plane compromising its integrity and inevitably crashing, due to extreme forces acting on parts of the plane that can't handle it.

Mass and size of each individual plane alters the forces acting upon them, inevitably affecting the acceleration and velocity of that plane. Mass in FSX is a changing variable, do to fuel consumption, however size is not so the resistance forces do not change but the acceleration force does; **F =ma,** therefore as mass changes but force doesn't, acceleration increases.

Gravity is always a dominant force when dealing with real life, and is therefore incorporated in FSX. Without gravity mass is then neglect able, due to the total energy of an object being the sum of kinetic and potential energy; potential energy relies on gravity otherwise it does not exist, and without it realism is impossible.

Collision is a very important aspect when simulating realism, because without it a sense of consequence is not established. Collision within FSX is very important, especially when planes are landing and taking off. A force of equal and opposite direction needs to be applied when an object collides with something (given that the object does not break apart), affecting the resulting new force on that object. When colliding with an object FSX also takes into account what part of the plane the collision has occurred with, providing an appropriate resistant force to it. Also affecting that section of the plane, which no longer operates at optimal efficiency, resulting in a change of net force.

Flight Simulator X is the closest example of real life flying to this date, as it accounts for almost all forces that act on the plane. However collision is not fully believable, apart from that FSX is a fairly good representation of piloting a plane.

Comparison Table:

	Stunt Pilot -Flash-	**Bump Copter 2 -Flash-**	**Flight Simulator X -PC-**	***Stunt Pilot 3D* -PC-**
Gravity	x	-	X	X
Plane Mass	-	-	X	x
Fuel consumption	-	-	X	-
Realistic Acceleration	x	x	X	X
Realistic Collision	x	-	X	x
Plane Thrust force	x	-	X	X
Resistance forces	x	-	X	x
Realistic controls	-	-	X	-
Realism	x	-	X	x

Key: **X:** Very high **x:** Medium -: none or very low

Summary:

Each of the games reviewed had physics appropriate to their platform and target audience.

Stun Pilot was for a general public and was designed to be fun and engaging, it did this by having players only control direction and only 2 speeds. By doing this players did not need to have an extensive knowledge of flying a plane and all the physics behind it, but simply fly the plane for fun. Collision was appropriately added to fit into gameplay, collide with anything and player looses, unrealistic by made sense.

Bump Copter 2 was designed as a kid's game to be fun, therefore physics within is had little to no effect. It suited its audience and platform by not being very processor hectic and providing a fun environment. Due to the lack of realistic physics players could 'bump' into anything and bounce off making it more enjoyable game.

Flight Simulator X is a realistic as possible game to simulate what it would be really like to fly a plane. The game had thought of exactly what is involved in flying a plane, mass, fuel consumption (affecting mass), acceleration, turbulence, wind resistance, streamlining

of planes, collision and had the necessary control panel to let players make informed decisions on how to fly properly. The physics engine is appropriate for the PC platform due to its multicores which could handle all the computations.

Stun Pilot 3D (my game) will comprise of real and surreal physics. Acceleration, gravity, mass and collision will try to mimic real life; this will involve a fairly accurate physics engine, appropriate to the platform. But also *Stunt Pilot 3D* is to be fun, therefore the complex control scheme's for flying a plane is taken out and replaced with a simpler scheme, and some physics will be tampered with to make the game more enjoyable to play.

Stunt Pilot 3D (First Person)

-PC-

The idea:

Stunt Pilot 3D is a first person flight simulator where players are required to navigate their plane through a small obstacle course of rings. The objective is to fly through the rings, to beat the clock in order to get bonus point. The points distributed to the player when flying through the rings is multiplied by the speed at which they travel. There is a variety of planes to choose from, each with unique characteristics of mass, thrust power, size and stability; these affect acceleration and handling for the planes.

Physics:

Stunt Pilot 3D will be using Newtonian physics to determine how fast a plane travels. The forces that are involved will be gravity, thrust, resistance forces and collision.

Gravity affects the forces on all planes; it is directly linked to potential gravitational energy formula and determines the thrust power needed for a plane to stay in the air.

Acceleration will be affected by how much thrust the player givens the engines of the plane. **F = ma,** by maximising the net force through thrust, and assuming mass stays a constant, acceleration increases. Acceleration also depends on what angle the plane is to the horizon, if the plane is facing down **a = g** x **sinθ,** this is the force getting applied to the plane due to gravity. Players within the game will have an accelerate and brake buttons, when held down it will apply the appropriate forces and once desired velocity is reached and the player release the button the plane will stay at that thrust power until another force is applied.

The normal/resistance forces will be as accurate as possible in *Stunt Pilot 3D,* simulating real life forces that act upon planes while in flight. However, *Stunt Pilot 3D* will not have weather simulations or varying air traffic environments, basic either day time or night time flight. This will allow for PC's that are not completely up to date with technology to

still run at optimum framerate, reducing the computation needed to run. The resistant forces include, drag, friction and normal force. All these forces depend on the plane that is being flown, depending on size and streamlining of the plane. A short overview with statistics will be given to the player about that plane that they are choosing when selecting a plane to play with.

Collision detection will be very basic, once a pilot collides with an object, control of the plane is instantly lost as the player ejects from the plane and watches it crash to the ground as power is cut to the engine. When the power is cut gravity takes over and the plane plummets to its death. Collision will again depend on the size and shape of the plane, every frame checking to see if anything is intercepting. Once a collision is detected the collision response will to simply cut all thrust and play the ‘eject player’ script.

References

'Collision Detection' 2009, Wikipedia, viewed 16 April 2009, http://en.wikipedia.org/wiki/Collision_detection

'Physics engine' 2009, Wikipedia, viewed 16 April 2009, http://en.wikipedia.org/wiki/Physics_engine

'Classical mechanics' 2009, Wikipedia, viewed 16 April 2009, http://en.wikipedia.org/wiki/Newtonian_physics

'Newton's laws of motion' 2009, Wikipedia, viewed 16 April 2009, http://en.wikipedia.org/wiki/Newtons_law

gregsometimes, 2007-09, 'Programming 3D Physics for Computer Games and Interactive Simulations', Authentic Society, viewed 16 April 2009, http://www.authenticsociety.com/about/3DPhysicsGames

Lawlor, T 2004, *The Physics of VideoGames*, Video Game Physics, viewed 16 April 2009, http://ffden-2.phys.uaf.edu/211_fall2004.web.dir/troy_lawlor/index.html

Purcell, L 2009, 'Microsoft Flight Simulator X SOARS to New Heights with Multi-Threading', Intel, viewed 16 April 2009, http://software.intel.com/en-us/articles/microsoft-flight-simulator-x-soars-to-new-heights-with-multi-threading/

Microsoft. 26.10.06. Flight Simulator X

Rock Solid Arcade. 2007. Stunt Pilot

ArcadeTown. 2004-2005. Bump Copter 2