

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



## MOBILE GAME PROGRAMMING


GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

## Course Outcomes :



CSGG4024.1	Develop an understanding of Objective-C and Cocos2d	Knowledge Level 6
CSGG4024.2	Apply advanced 2D Graphics and designing for mobile	Knowledge Level 3
CSGG4024.3	Make use of mobile usability and design concerns	Knowledge Level 3
CSGG4024.4	Understand advanced graphical and audio effects	Knowledge Level 2
CSGG4024.5	Understand three dimensional concept in mobile environment	Knowledge Level 2
CSGG4024.6	Implement individual game project or in a team environment	Knowledge Level 5

## Course Prerequisites

**The objective of this course is to:**

- Provide the overview of professional game design
- Know the best ways to learn how to design games
- Understand what game design is and isn't

## Course Description

- This course provides students with an in-depth introduction to technologies and techniques used to create successful mobile games.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



**Texture Atlases**

GALGOTIAS  
UNIVERSITY

## Texture atlases

- Texture atlases for 2D games is a great optimization for batching together tons of different sprites (especially quads) with a very few number of [draw calls](#). Making them is a pain. For pixel art or other 2D art, [DXT compression](#) is often not a great choice [due to the lossyness](#). One good way to make atlases for 2D games is with the PNG format. Single-file loaders and savers can be used to load and save PNGs with a single function call each.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

Here's an example with the popular [stb\\_image](#), along with [stb\\_image\\_write](#):

- `#define STB_IMAGE_IMPLEMENTATION`
- `#include "stb_image.h"`
- `#define STB_IMAGE_WRITE_IMPLEMENTATION`
- `#include "stb_image_write.h"`
- `int w;`
- `int h;`
- `int comp;`
- `unsigned char* image = stbi_load( path, &w, &h, &comp, STBI_rgb_alpha );`
- `stbi_write_png( path, w, h, comp, image, 4 );`

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- This requires two different headers, one for loading and one for saving. Additionally the author of these headers also has a bin packing header, which could be used to make a texture atlas compiler.
- However I have created a single-file header called [tinydeflate](#). It does PNG loading, saving, and can create a texture atlas given an array of images. Internally it contains its own bin packing algorithm for sorting textures and creating atlases. Here's an example:

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- #define TINYDEFLATE\_IMPL
- #include "tinydeflate.h"
- const char\* names = {
- "path/image0.png",
- "path/image1.png"
- };
- tdlmage img0 = tdLoadPNG( names[ 0 ] );
- tdlmage img1 = tdLoadPNG( names[ 1 ] );
- int w = 64;
- int h = 64;
- int count = 2;
- tdMakeAtlas( "atlas.png", "atlas.txt", w, h, pngs, count, names );

GALGOTIAS  
UNIVERSITY

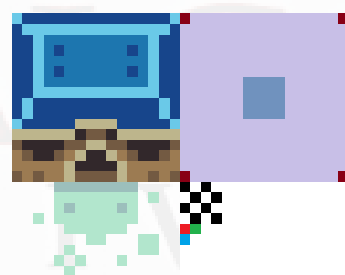


# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- The above example (if given a few more images) could output an atlas like so:



GA  
UN  
AS  
TY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

Followed by a very easy to parse text formatted file containing uv information for each atlas:

```
atlas.png
8
{ "imgs/1x1.png", w = 16, h = 32, u = { 0.0000000000, 0.4998779297 }, v = { 0.24987792
{ "imgs/4x4.png", w = 16, h = 32, u = { 0.0000000000, 0.9998779297 }, v = { 0.24987792
{ "imgs/debug_tile.png", w = 16, h = 32, u = { 0.2500000000, 0.4998779297 }, v = { 0.4
{ "imgs/default.png", w = 18, h = 20, u = { 0.5000000000, 0.3123779297 }, v = { 0.7811
{ "imgs/house_blue.png", w = 16, h = 16, u = { 0.2500000000, 0.7498779297 }, v = { 0.4
{ "imgs/house_red.png", w = 4, h = 4, u = { 0.2500000000, 0.8123779297 }, v = { 0.3123
{ "imgs/house_yellow.png", w = 2, h = 2, u = { 0.2500000000, 0.8436279297 }, v = { 0.2
{ "imgs/squinkle.png", w = 1, h = 1, u = { 0.2812500000, 0.8280029297 }, v = { 0.29675
```

UNIVERSITY

## What's a Texture Atlas?

- If you are approaching the development of a 3D video game for the first time, you'll begin to discover that 3D graphics are composed of several parts: 3D meshes, textures, particle systems, and many other elements that are usually drawn on the screen 30 times per second (in slang: 30 fps) during the rendering process, making the game's world varied and lively.
- Believe it or not, the first 3D video games I saw in my life had none of these elements. They were composed only of lines that formed objects or elements in 3D wireframe.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



Name of the Faculty: Mr.Karthick.R

Program Name: B.Tech(Spl)

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- In a 3D game, the UI is usually made of 3D elements (such as planes or boxes) with textures.
- We mentioned before the rendering process: it's the operation by which the elements in memory are physically drawn on the screen. It's among the most complex and expensive processes that occur in a real-time 3D game. Then, any expedient to reduce the time taken by this process is welcome; less time spent in the rendering phase means a higher frame rate (i.e. if you reach the 60 fps you can render the image twice and then think of developing your game also for VR), or more screen elements (and then a richer game, more animated, more beautiful).
- One of the means used to reduce the duration of the rendering process is a Texture Atlas: it's nothing more than an image that contains many textures.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



Name of the Faculty: Mr.Karthick.R

Program Name: B.Tech(Spl)



## How a Texture Atlas Works

GALGOTIAS  
UNIVERSITY

## How a Texture Atlas Works

- Note: As mentioned in the previous paragraph, this article will discuss the Texture Atlas applied to the UI. However, many of the concepts explained here can also be applied to 3D models and their textures.
- A Texture Atlas, we said, is a collection of textures inside a single image.

GALGOTIAS  
UNIVERSITY

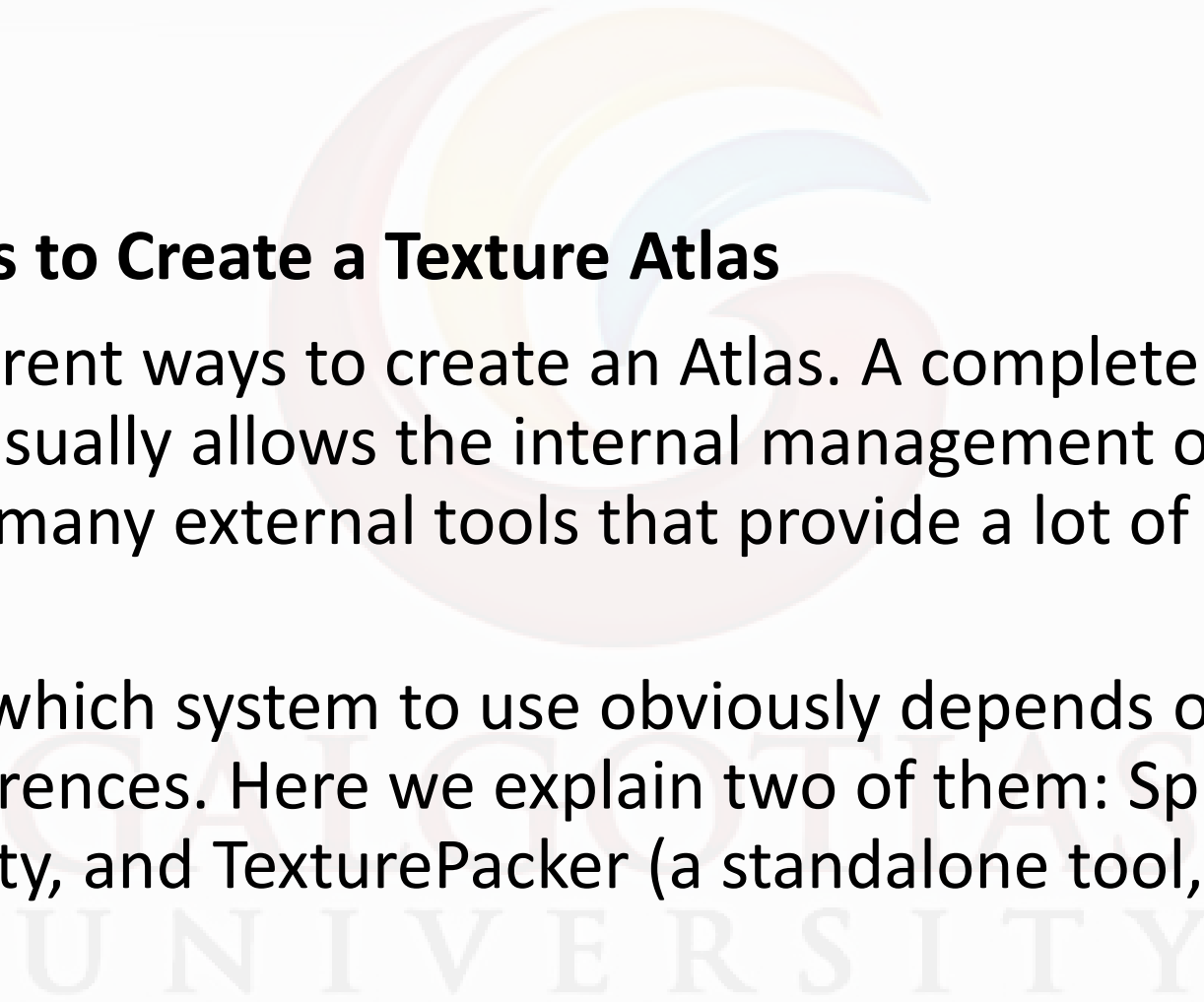


## How a Texture Atlas Works

- An Atlas is usually associated with a file descriptor, which indicates to the game where a texture is (in certain x and y coordinates), in order to retrieve it.
- Depending on the system that you will use to generate and manage the Atlas, you will have more or less options, such as the distance between the images that compose it (reducing the risk of artifacts on the edges of the texture, caused by an overlap of two elements), or the ability to rotate the elements to optimize the space inside the Atlas (more optimized space means more images inside the same Atlas).

GALGOTIAS  
UNIVERSITY

- **Different Ways to Create a Texture Atlas**
- There are different ways to create an Atlas. A complete development environment usually allows the internal management of the Atlas; there are also many external tools that provide a lot of additional options.
- The choice of which system to use obviously depends on your personal preferences. Here we explain two of them: Sprite Packer, internal to Unity, and TexturePacker (a standalone tool, for a fee).



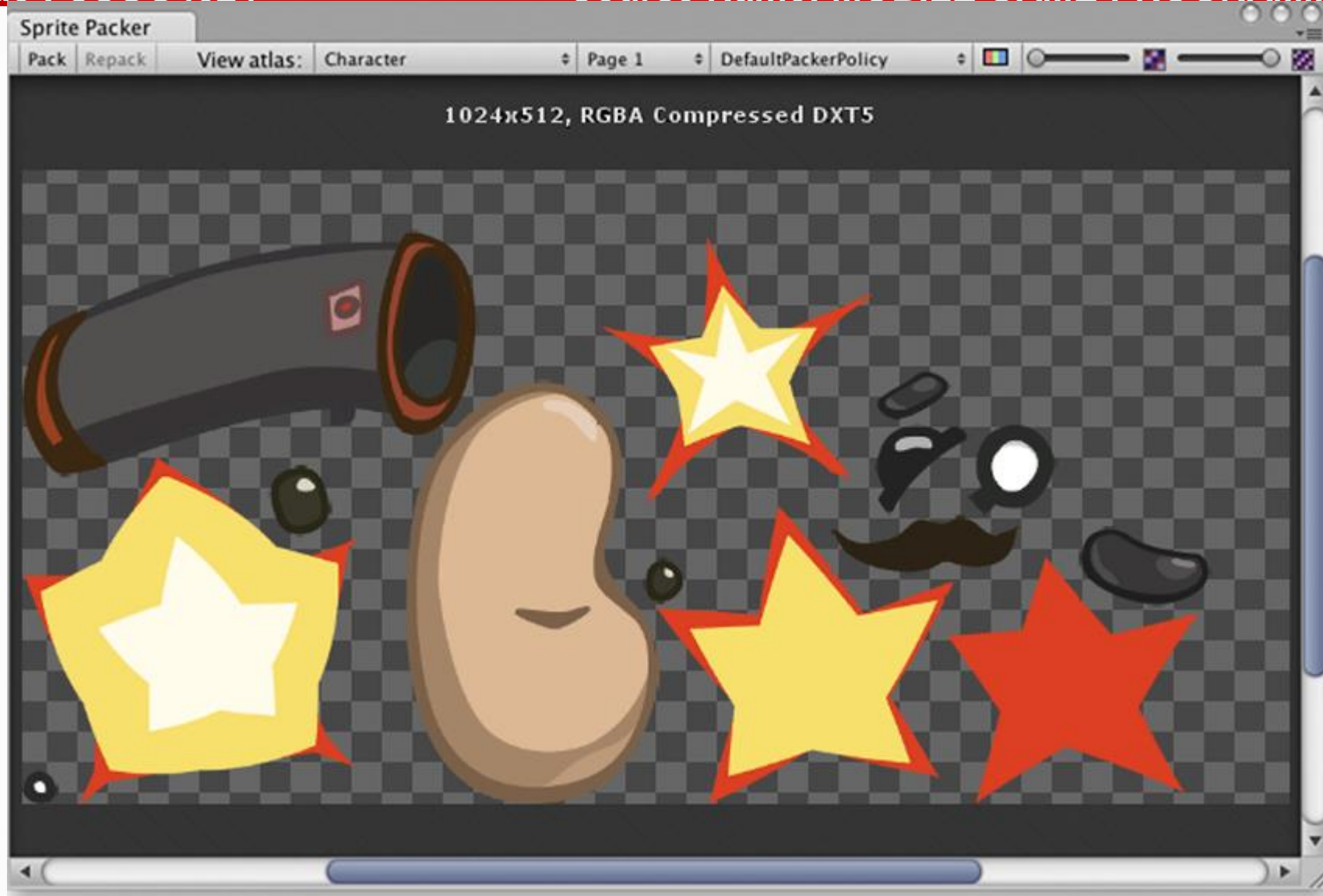
## Sprite Packer

- To open Sprite Packer, choose from the menu **Window > Sprite Packer**.
- The management is really easy: the button Pack is used to create one or more Atlases (it depends on the number of your images and on the Atlas dimension that you want to use).
- Now you can select an image to see where it is in the Atlas. If you add or remove images from your project, you must use the button Repack, to update the Atlas.
- In order to configure the Sprite Packer, you can choose from the menu **Edit > Project Settings > Editor**; here you can disable the Atlas, activate it only for the game built, or always turn it on.

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



Name of the Faculty: Mr.Karthick.R

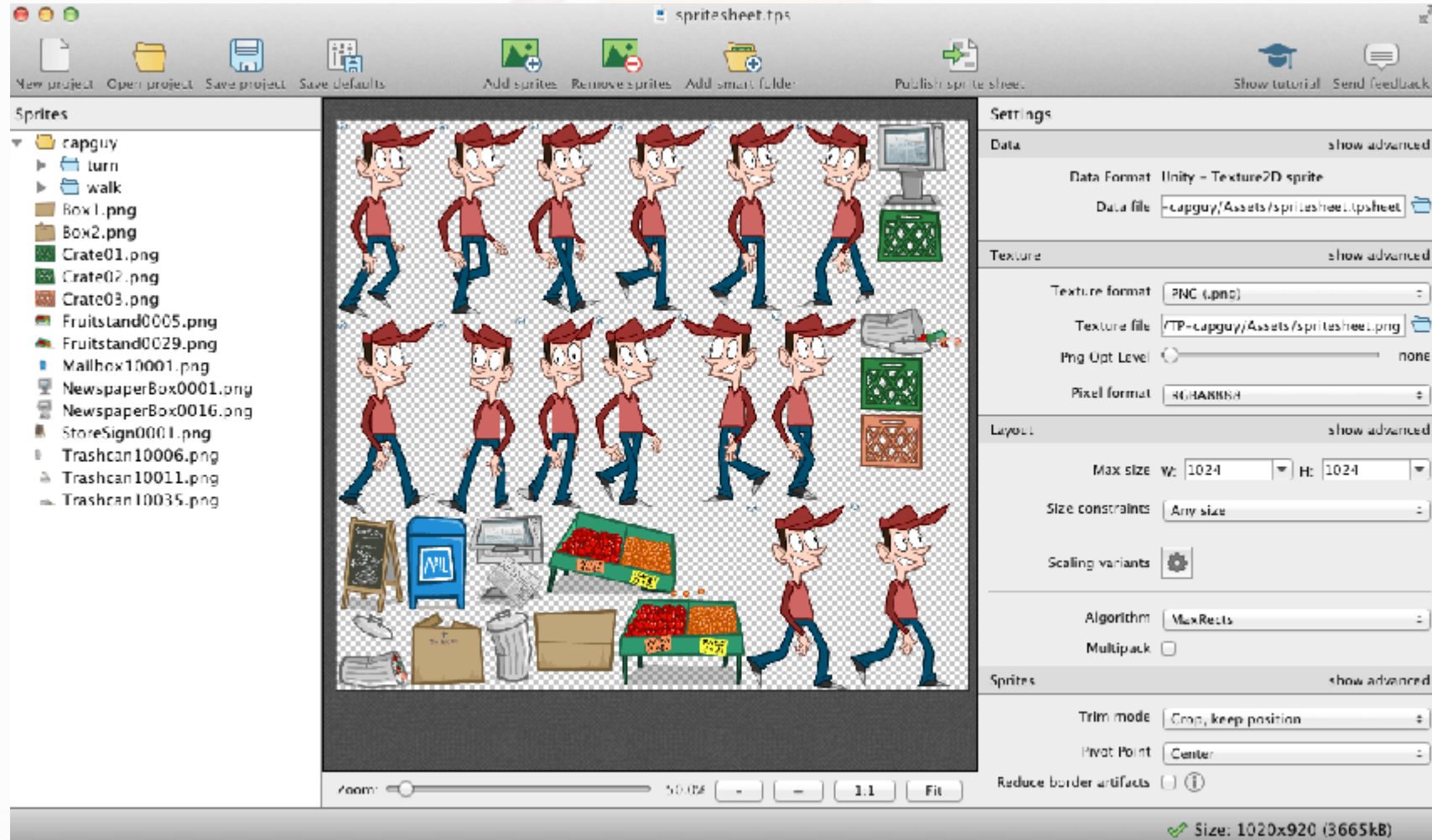
Program Name: B.Tech(Spl)

- **Texture Packer**
- Texture Packer is a standalone tool used to manage Atlas.
- You can add one or more folders from your project and Texture Packer will create the Atlas.
- After that, you can choose the data format for the export. As you can see, there is also the option "JSON for Unity". This means that you can export your Atlas for your Unity project. But, in order to use them together, you must install a free editor extension from [the asset store](#).

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



## Why Is It Important to Use a Texture Atlas?

- But why is it so important to collect multiple images into a single larger one?
- Let's go back for a moment to the rendering process: if every element of the UI has a separate texture, it is drawn with a separated "draw call." This means that if in our interface we have the icon of hearts (representing the player's energy) and the icon of the coins collected, we will have two draw calls.
- Each draw call takes some time to complete, making the rendering process longer and longer. If there are five UI elements, instead of two as in the example above, there are five draw calls.

GALGOTIAS  
UNIVERSITY

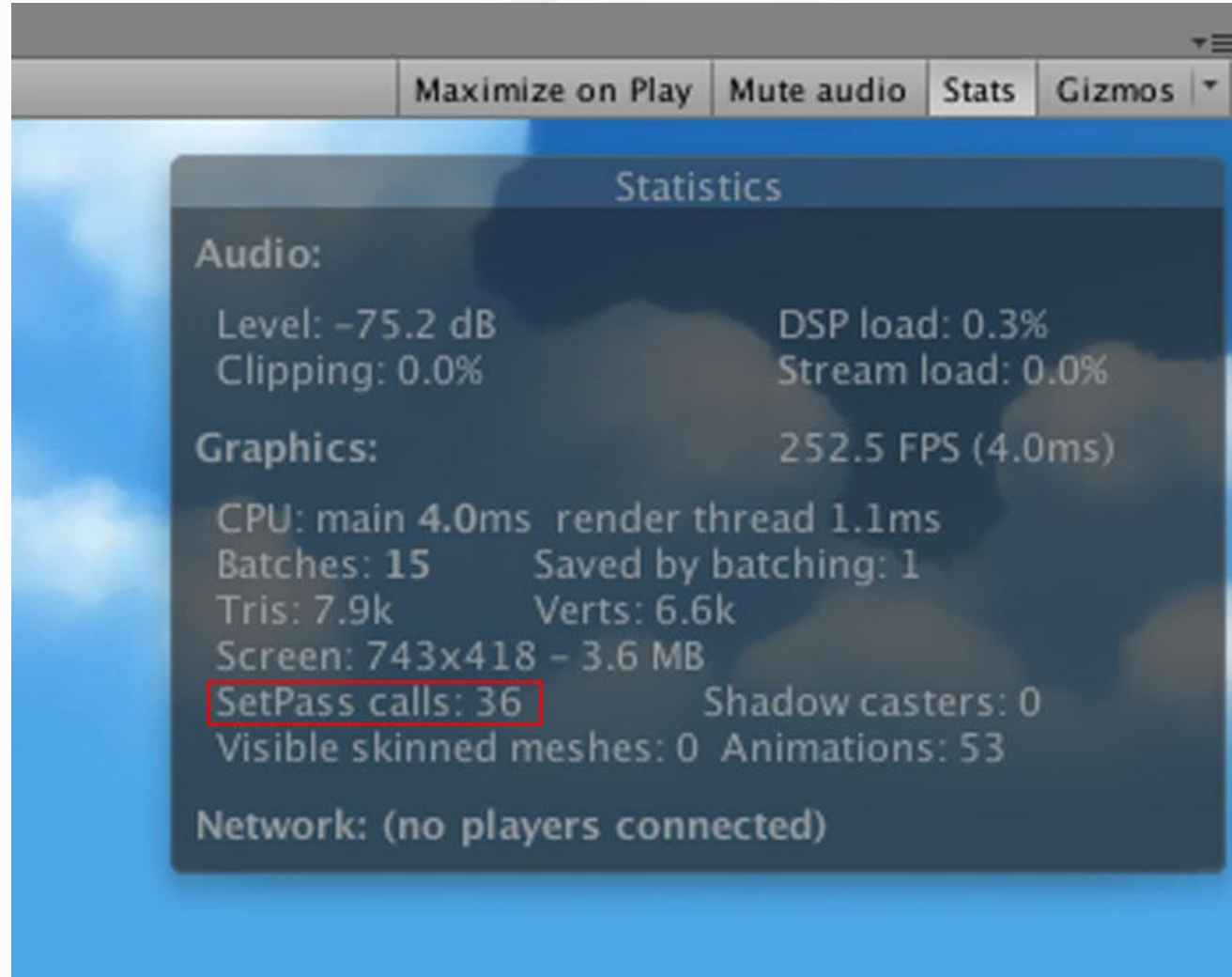
- Do you begin to see the point?
- More draw calls -> more time during the rendering phase -> less fps -> game with a low frame rate (with some frame drops) or fewer elements on the screen (then visually poor).
- Wasting draw calls this way, unless there are special reasons, doesn't really make sense, especially for the UI.
- In fact, all the textures in an Atlas will be rendered together, in a single pass.



# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- In conclusion, especially if you're developing a game on a platform where performance is really important (such as a mobile platform):
- You must pay attention to the number of draw calls: more draw calls means a higher rendering time (and a higher rendering time means the risk of having a low frame rate).
- Generally, every object with a different texture can generate a single draw call (it's a generic statement: there are some exceptions, especially in the case of 3D objects).
- One way to lower the number of draw calls is to use a Texture Atlas.
- A Texture Atlas is basically a big texture with a group of different textures.
- All objects that use the same Texture Atlas generate a single draw call.
- Especially for the UI textures, the use of a Texture Atlas is a must-have to improve the performance of your project.

## Importing Textures

- There are 3 ways you can import texture assets into PlayCanvas:
- Drag and drop images into the Assets panel.
- Select 'Upload' from the context menu in the Assets panel and select an image using the file browser.
- Import an FBX file that embeds textures.
- Supported image formats are:
  - JPG
  - PNG
  - GIF
  - TGA
  - BMP
  - TIF
  - HDR
  - EXR

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- Imported JPG and PNG files remain in their original format.
- GIF, TGA, BMP and TIF image types will be converted to JPG or PNG on import. If the imported image has transparency, it will be converted to PNG. Otherwise, it will be converted to JPG.
- HDR and EXR are [high dynamic range formats](#) formats. Images of these types are converted to PNG on import and marked as being stored in RGBM format. RGBM essentially stores a multiplier for RGB values in the PNG's alpha channel, enabling the compression of an HDR format into a low dynamic range format.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- By default, imported images will be resized to the nearest power of two. For example, an image that is 323x414 will be resized to 256x512 on import. This is done because the graphics engine cannot utilize mipmapping with non-power of two textures. However, this behavior can be overridden by disabling the 'Textures POT' setting in the Asset Tasks panel before importing a non-power of two texture.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



**Texture Properties**

GALGOTIAS  
UNIVERSITY

## Texture Properties

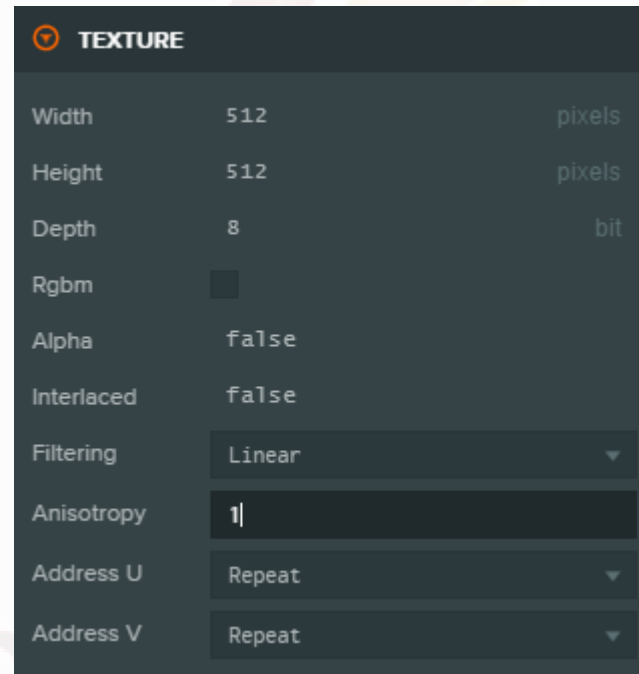
- Selecting a texture's thumbnail in the Assets panel will load it into the Inspector panel. Note that you can multi-select textures and edit the whole selection simultaneously in the Inspector.
- A texture shares the standard set of asset properties (ID, name, tags and so on). But it's also has some texture-specific properties.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



GALGUTIAS  
UNIVERSITY



## Texture Filtering

- Texture filtering gives control over how the color of a texture mapped pixel is calculated. 'Point' applied no filtering whereas 'Linear' will interpolate the color of a texel with those of its neighbours. This produces better visual results, particularly as a texture is minimized (where the texture occupies fewer pixels on the screen than it has texels).

GALGOTIAS  
UNIVERSITY

## Anisotropy

- When textures are viewed on surfaces at an oblique angle, quality can suffer and they can appear blurred. To fix this problem, you can set a value for anisotropy. See how different anisotropy values can affect the appearance of a texture:

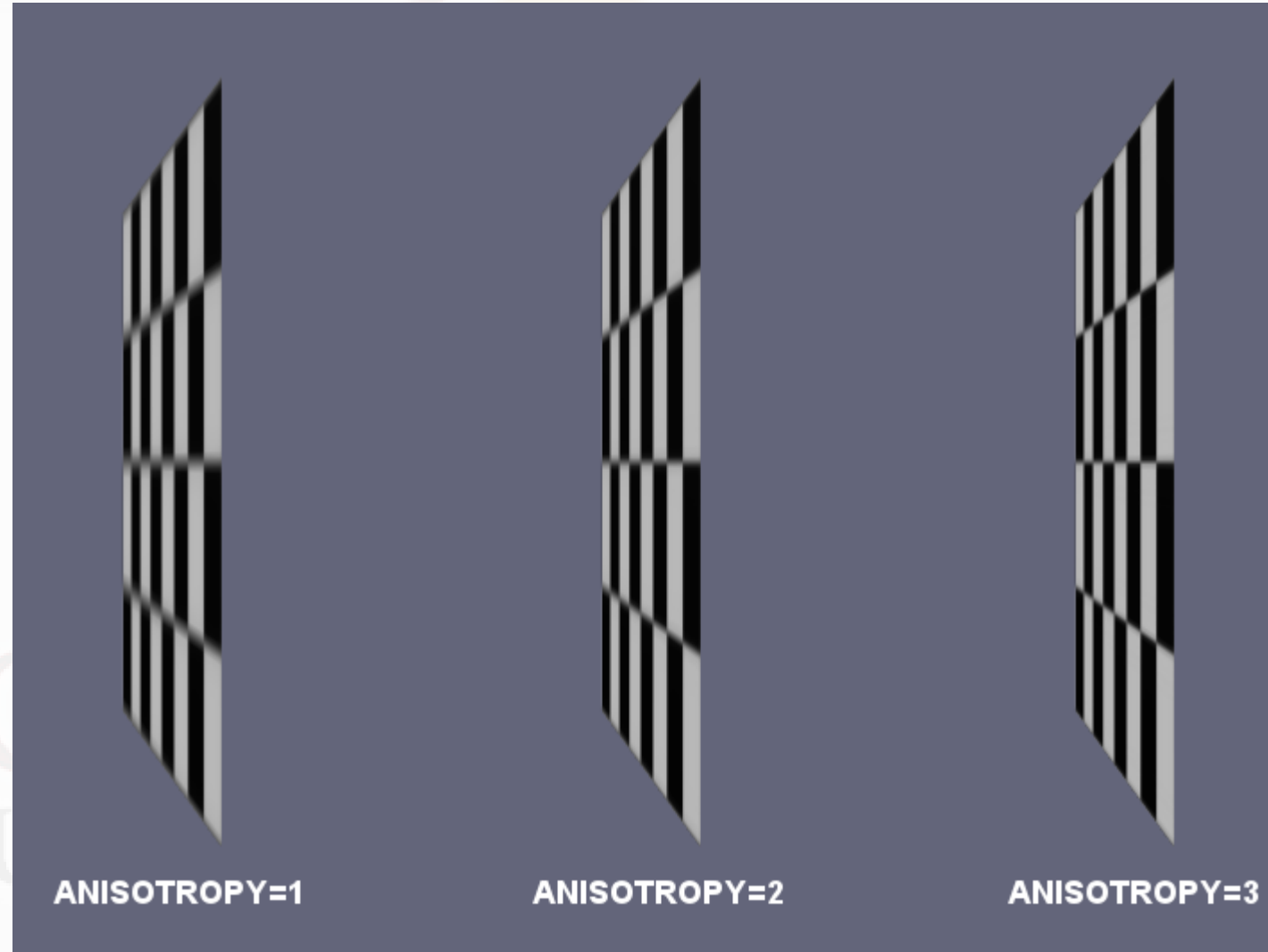
GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

Note that as anisotropy increases, the cost of sampling the texture on the GPU also increases.



## Texture Addressing

- The texture addressing properties give you control over how a texture is sampled for texture coordinates outside the range 0 to 1. See how the different modes affect the sprite below:

GALGOTIAS  
UNIVERSITY

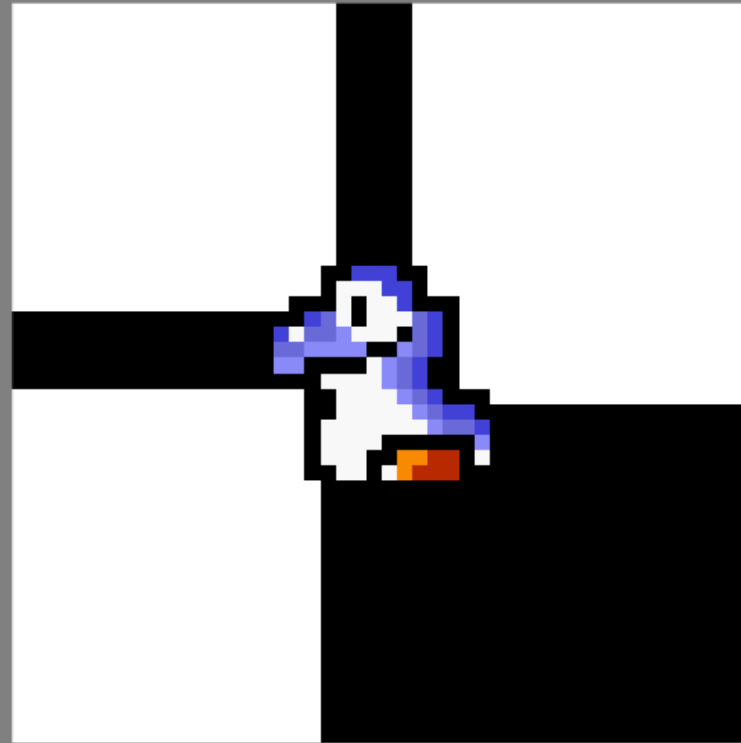
# School of Computing Science and Engineering

Course Code : CSGG4024

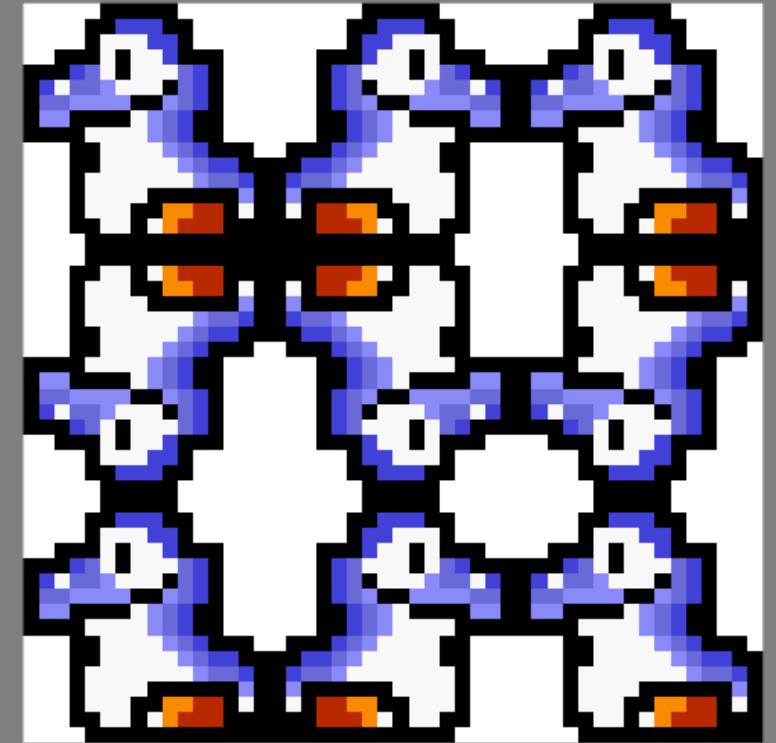
Course Name: MOBILE GAME PROGRAMMING



REPEAT



CLAMP



MIRROR

## Texture Compression

- Texture data is stored in a device's video memory (or VRAM). It is important to ensure that your application does not exhaust VRAM as this can cause undesirable things like browser tab crashes.
- The Editor has the ability to apply lossy compression schemes to your textures to dramatically reduce the amount of VRAM used. These schemes are:
  - DXT: Typically supported by desktop devices.
  - PVR: Typically supported by iOS devices.
  - ETC: Typically supported by Android devices.

GALGOTIAS  
UNIVERSITY

Consider this texture asset



# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- It's a 512x512 JPG that is 202KB in size. However, JPG is a compressed format and when passed to the graphics engine, it is expanded to an uncompressed RGB8 format that occupies 1.05MB of VRAM (including mipmap levels).
- Enabling all compression schemes achieves the following results:

The compression has achieved a 6 times reduction in VRAM usage. Furthermore, in this case, compression has also reduced download size from 202KB to as little as 116KB.

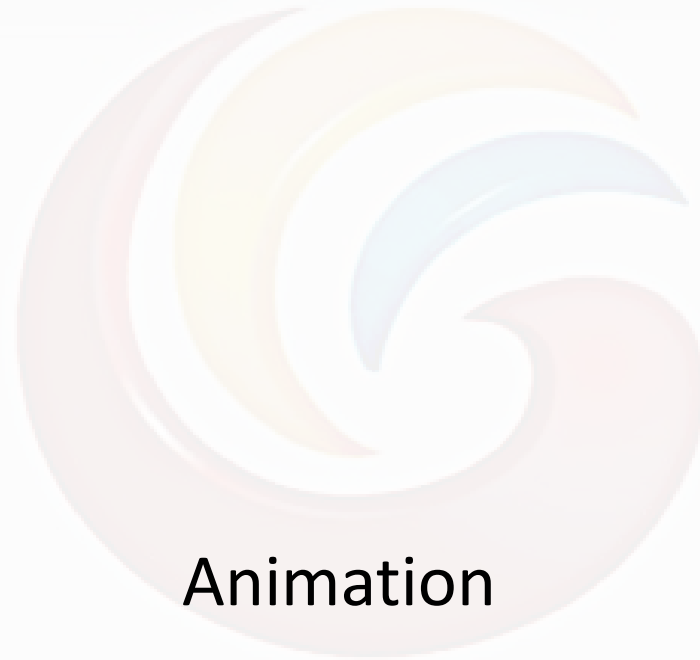




# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



Animation

GALGOTIAS  
UNIVERSITY

## Animation

- It is a method of photographing successive drawings, models, or even puppets, to create an illusion of movement in a sequence. Because our eyes can only retain an image for 1/16 of a second, when multiple images appear in fast succession, the brain blends them into a single moving image. |
- In traditional animation, pictures are drawn or painted on transparent celluloid sheets to be photographed and shown on film.
- Early cartoons are examples of this, but today, most animation is made with computer-generated imagery or CGI.

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- To create the appearance of smooth motion from these drawn, painted, or computer-generated images, frame rate, or the number of consecutive images that are displayed each second, is considered. Moving characters are usually shot “on twos” which just means one image is shown for two frames, totaling in at 12 drawings per second. 12 frames per second allows for motion but may look choppy. In the film, a frame rate of 24 frames per second is often used for smooth motion animation.
- There are several types of animation that employ different techniques to achieve their desired effect.

GALGOTIAS  
UNIVERSITY

## Different Types of Animation:

- Traditional Animation
- 2D Animation (Vector-based)
- 3D Animation
- Motion Graphics
- Stop Motion

The logo of Galgotias University is a stylized 'G' composed of three curved, overlapping bands in shades of yellow, blue, and red. Below the logo, the text 'GALGOTIAS UNIVERSITY' is written in a large, light grey, serif font.

GALGOTIAS  
UNIVERSITY



Designing for the impatient user

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- 1. Equip with all possible information
- 2. Never lengthen a straight-forward flow
- 3. Avoid heavy-duty components

The logo of Galgotias University is a stylized 'G' composed of three curved, overlapping bands in shades of yellow, blue, and pink. Below the logo, the text 'GALGOTIAS UNIVERSITY' is displayed in a large, light grey, serif font, with 'GALGOTIAS' on the top line and 'UNIVERSITY' on the bottom line.

GALGOTIAS  
UNIVERSITY

## Quick overview of vector math

- **Vector**, in [mathematics](#), a quantity that has both magnitude and direction but not position. Examples of such quantities are [velocity](#) and [acceleration](#).
- Vectors may be visualized as directed [line](#) segments whose lengths are their magnitudes. Since only the magnitude and direction of a [vector](#) matter, any directed segment may be replaced by one of the same length and direction but beginning at another point, such as the origin of a [coordinate](#) system.

GALGOTIAS  
UNIVERSITY

## Physics principles

- We start from the following five basic principles to construct all other physical laws and equations. These five basic principles are:
- (1) Constituent principle: the basic constituents of matter are various kinds of identical particles. This can also be called locality principle;
- (2) Causality principle: the future state depends only on the present state;
- (3) Covariance principle: the physics should be invariant under an arbitrary coordinate transformation;
- (4) Invariance or Symmetry principle: the spacetime is homogeneous;
- (5) Equi-probability principle: all the states in an isolated system are expected to be occupied with equal probability.

GALGOTIAS  
UNIVERSITY



# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

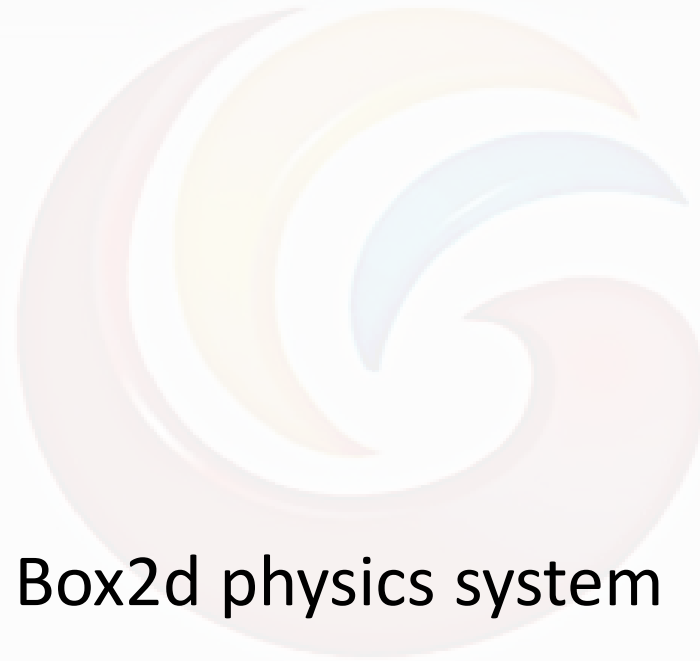
- These five basic principles can be considered as physical common senses. It is very natural to have these basic principles. More important is that these five basic principles are consistent with one another. From these five principles, we derive a vast set of equations which explains or promise to explain all the phenomena of the physical world.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



Box2d physics system

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- Box2D is a 2D rigid body simulation library for games. Programmers can use it in their games to make objects move in realistic ways and make the game world more interactive. From the game engine's point of view, a physics engine is just a system for procedural animation.
- Box2D is written in portable C++. Most of the types defined in the engine begin with the b2 prefix.

Hopefully this is sufficient to avoid name clashing with your game engine.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- Box2D works with several fundamental concepts and objects. We briefly define these objects here and more details are given later in this document.
- **shape**
- A shape is 2D geometrical object, such as a circle or polygon.
- **rigid body**
- A chunk of matter that is so strong that the distance between any two bits of matter on the chunk is constant. They are hard like a diamond. In the following discussion we use body interchangeably with rigid body.
- **fixture**
- A fixture binds a shape to a body and adds material properties such as density, friction, and restitution. A fixture puts a shape into the collision system (broad-phase) so that it can collide with other shapes.

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- **constraint**
- A constraint is a physical connection that removes degrees of freedom from bodies. A 2D body has 3 degrees of freedom (two translation coordinates and one rotation coordinate). If we take a body and pin it to the wall (like a pendulum) we have constrained the body to the wall. At this point the body can only rotate about the pin, so the constraint has removed 2 degrees of freedom.
- **contact constraint**
- A special constraint designed to prevent penetration of rigid bodies and to simulate friction and restitution. You do not create contact constraints; they are created automatically by Box2D.
- **joint**
- This is a constraint used to hold two or more bodies together. Box2D supports several joint types: revolute, prismatic, distance, and more. Some joints may have limits and motors.

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- **joint limit**
- A joint limit restricts the range of motion of a joint. For example, the human elbow only allows a certain range of angles.
- **joint motor**
- A joint motor drives the motion of the connected bodies according to the joint's degrees of freedom. For example, you can use a motor to drive the rotation of an elbow.
- **world**
- A physics world is a collection of bodies, fixtures, and constraints that interact together. Box2D supports the creation of multiple worlds, but this is usually not necessary or desirable.

SAIGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

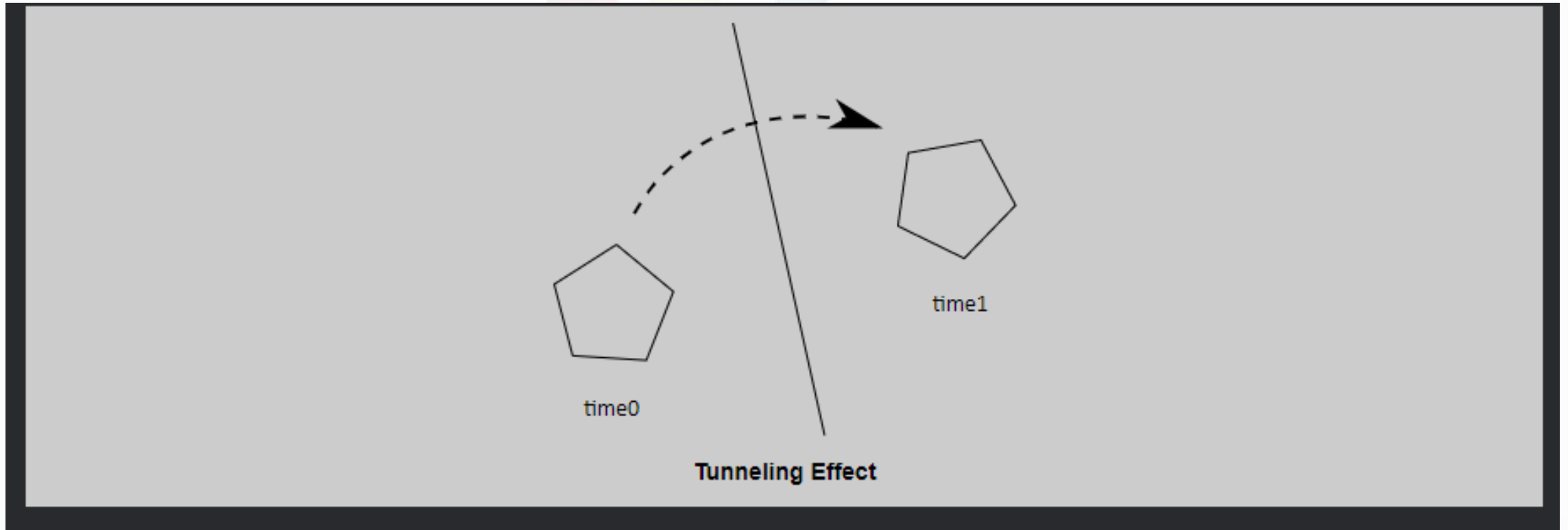
- **solver**
- The physics world has a solver that is used to advance time and to resolve contact and joint constraints. The Box2D solver is a high performance iterative solver that operates in order N time, where N is the number of constraints.
- **continuous collision**
- The solver advances bodies in time using discrete time steps. Without intervention this can lead to tunneling.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING





# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING

- Box2D contains specialized algorithms to deal with tunneling. First, the collision algorithms can interpolate the motion of two bodies to find the first time of impact (TOI). Second, there is a sub-stepping solver that moves bodies to their first time of impact and then resolves the collision.

GALGOTIAS  
UNIVERSITY

## Modules

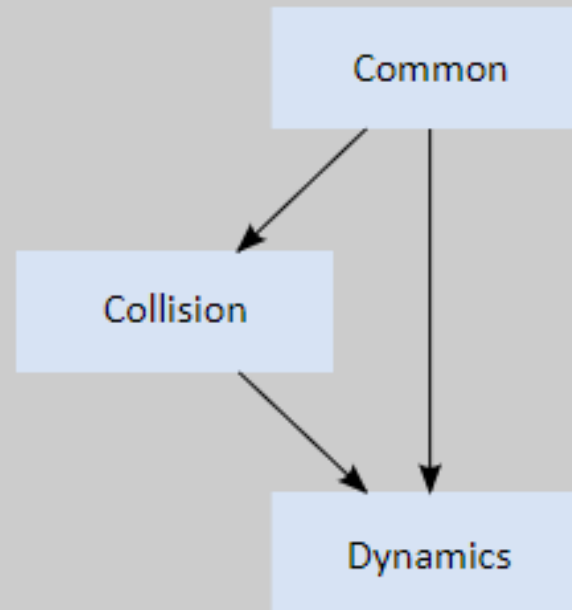
- Box2D is composed of three modules: Common, Collision, and Dynamics. The Common module has code for allocation, math, and settings. The Collision module defines shapes, a broad-phase, and collision functions/queries. Finally the Dynamics module provides the simulation world, bodies, fixtures, and joints.

GALGOTIAS  
UNIVERSITY

# School of Computing Science and Engineering

Course Code : CSGG4024

Course Name: MOBILE GAME PROGRAMMING



**Box2D Modules**



Thank You