# Efectos basados en imágenes

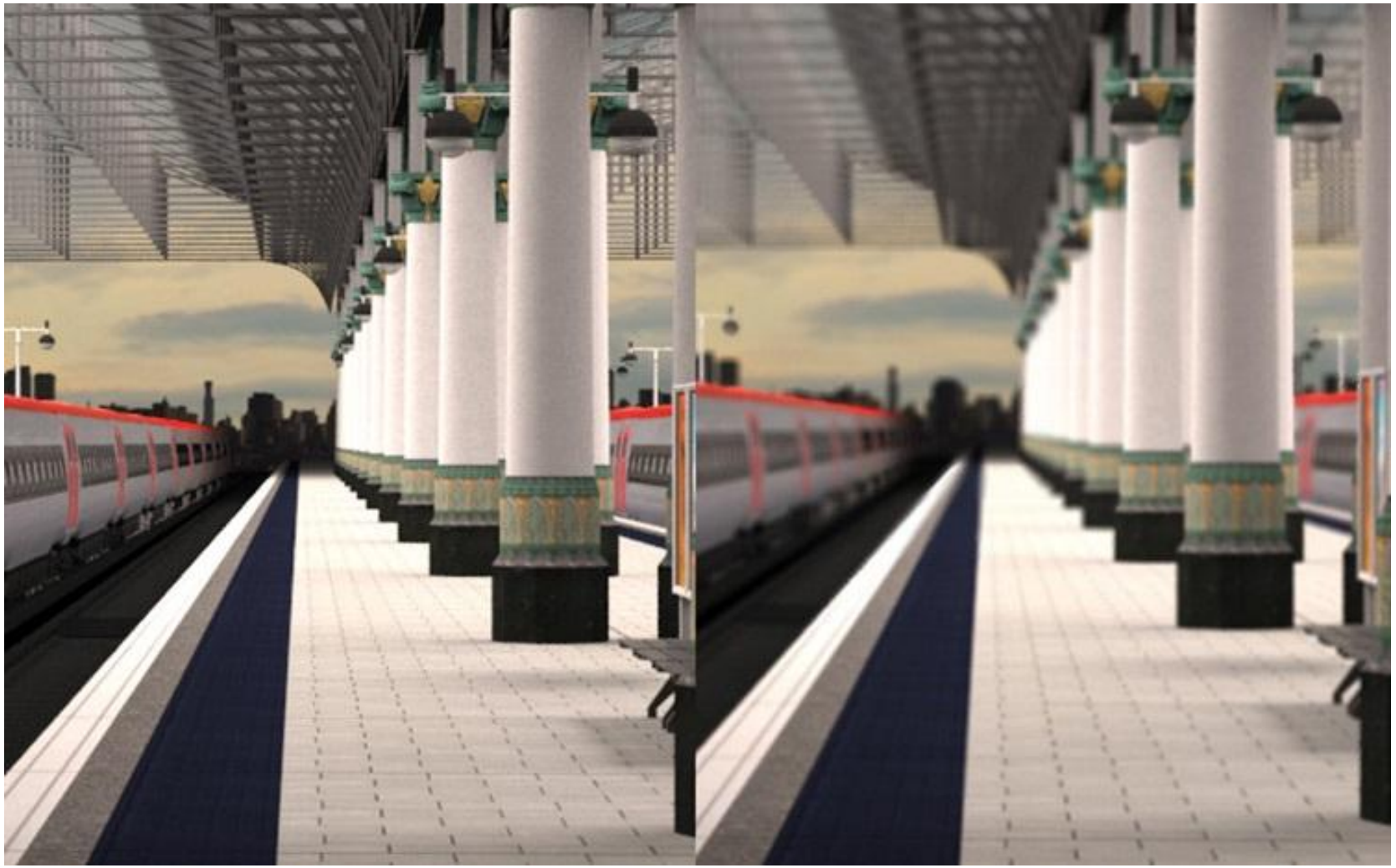
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Efectos basados en imágenes

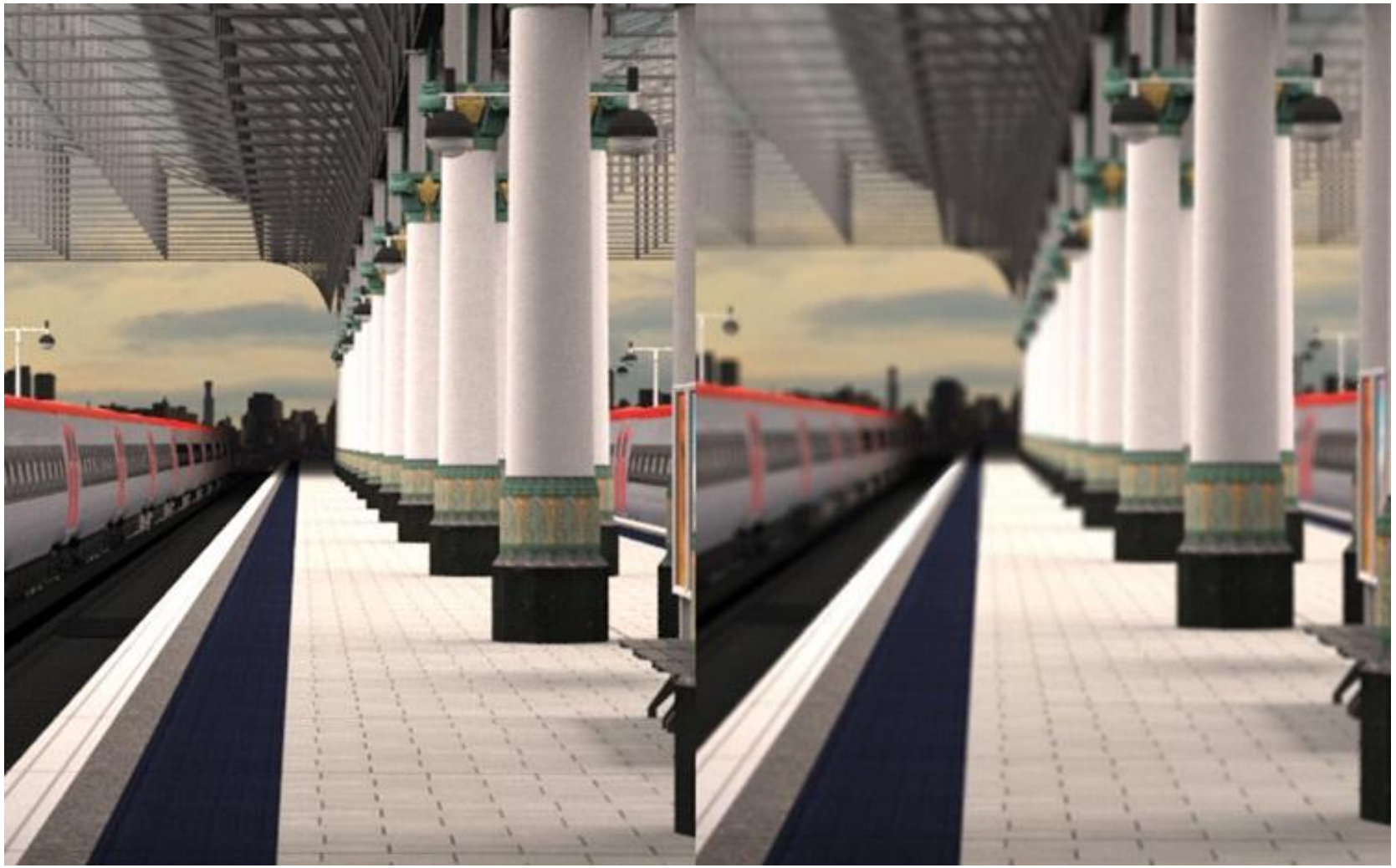
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

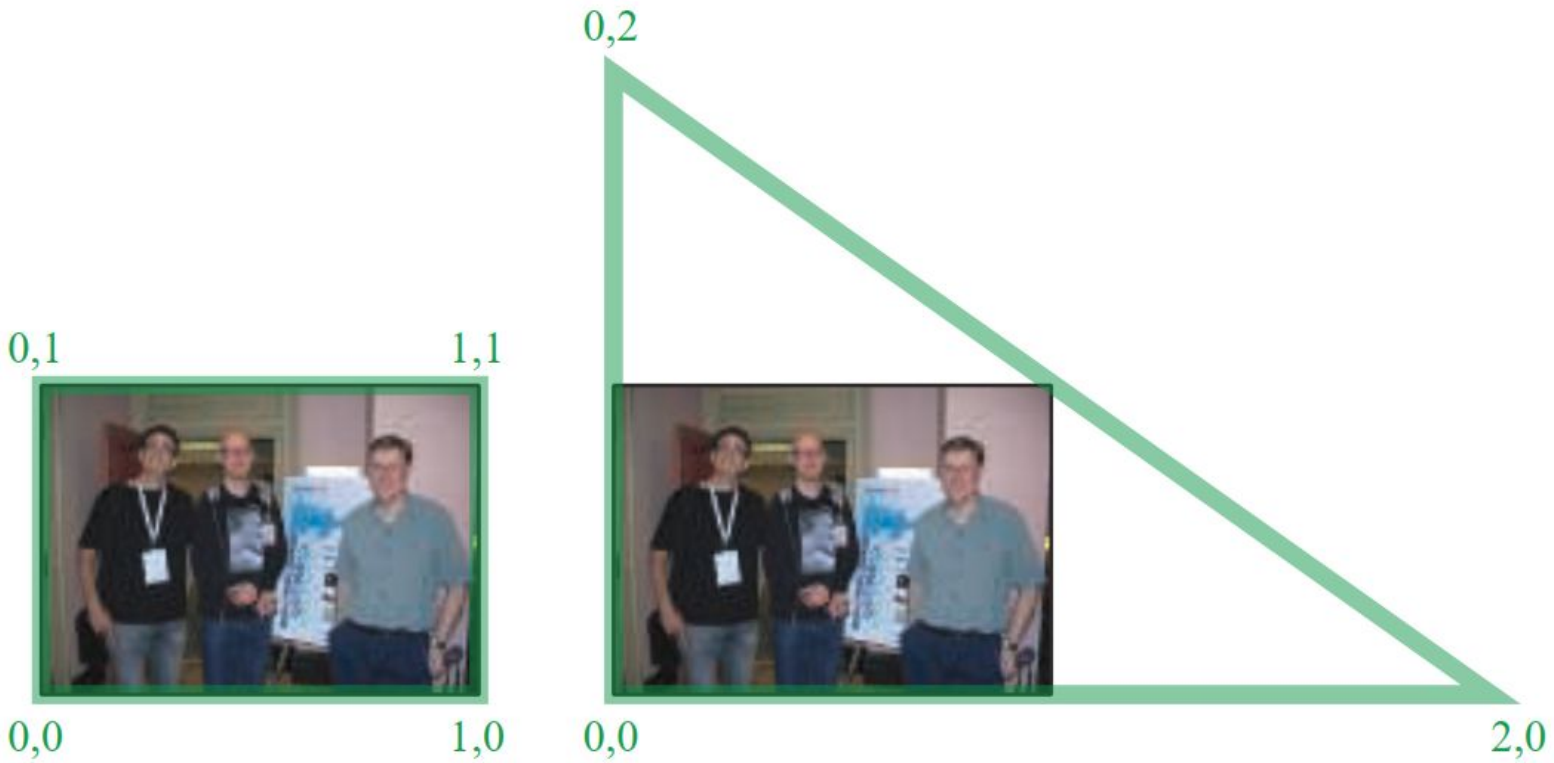
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.
Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

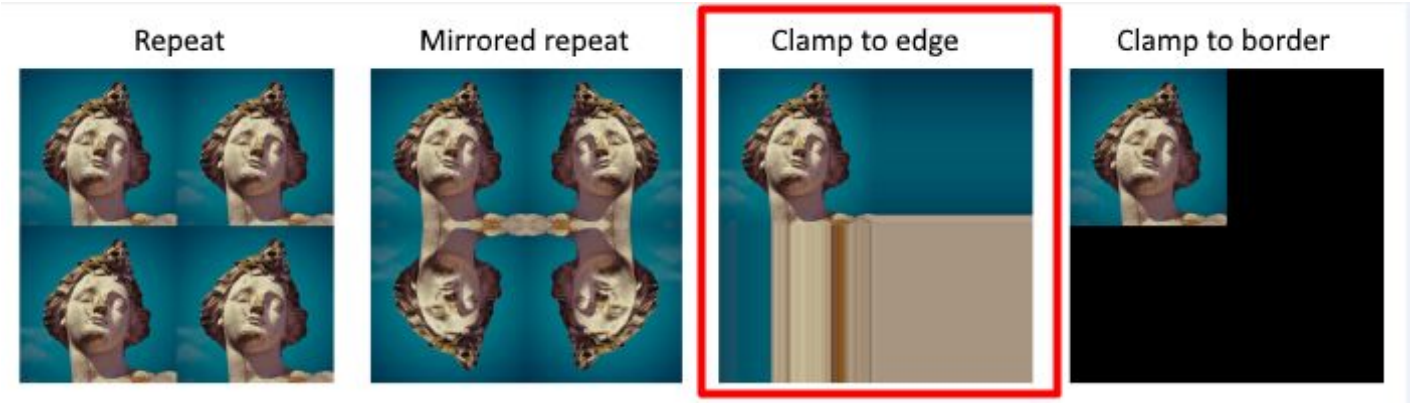
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

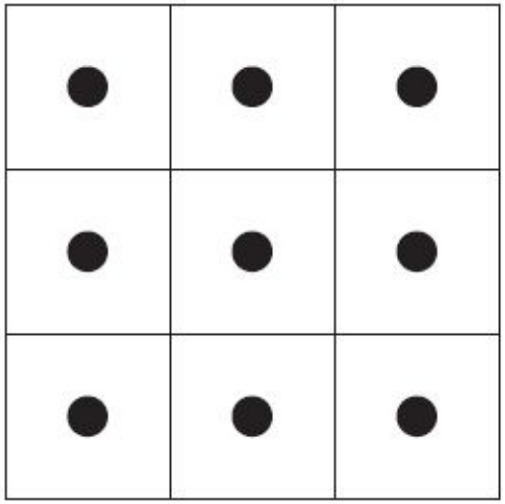
(c)

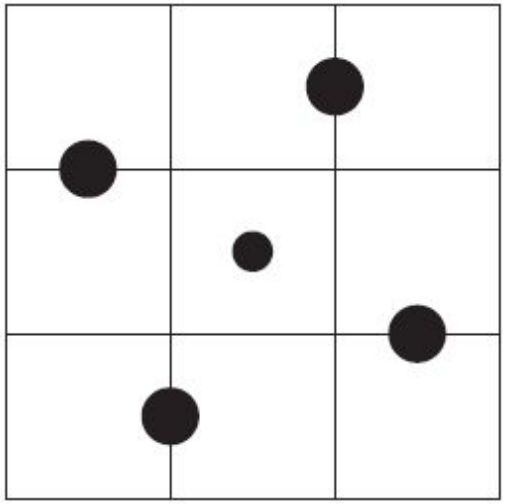
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

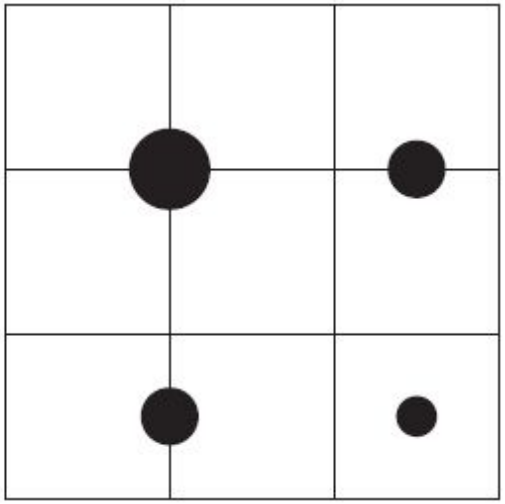
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

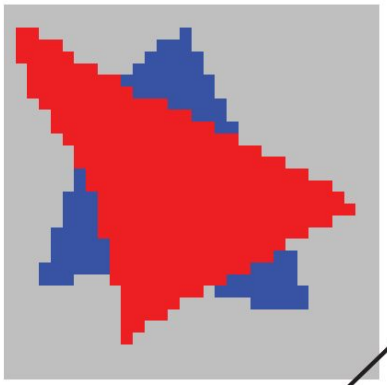
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

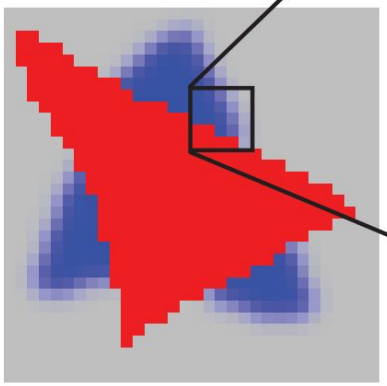
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

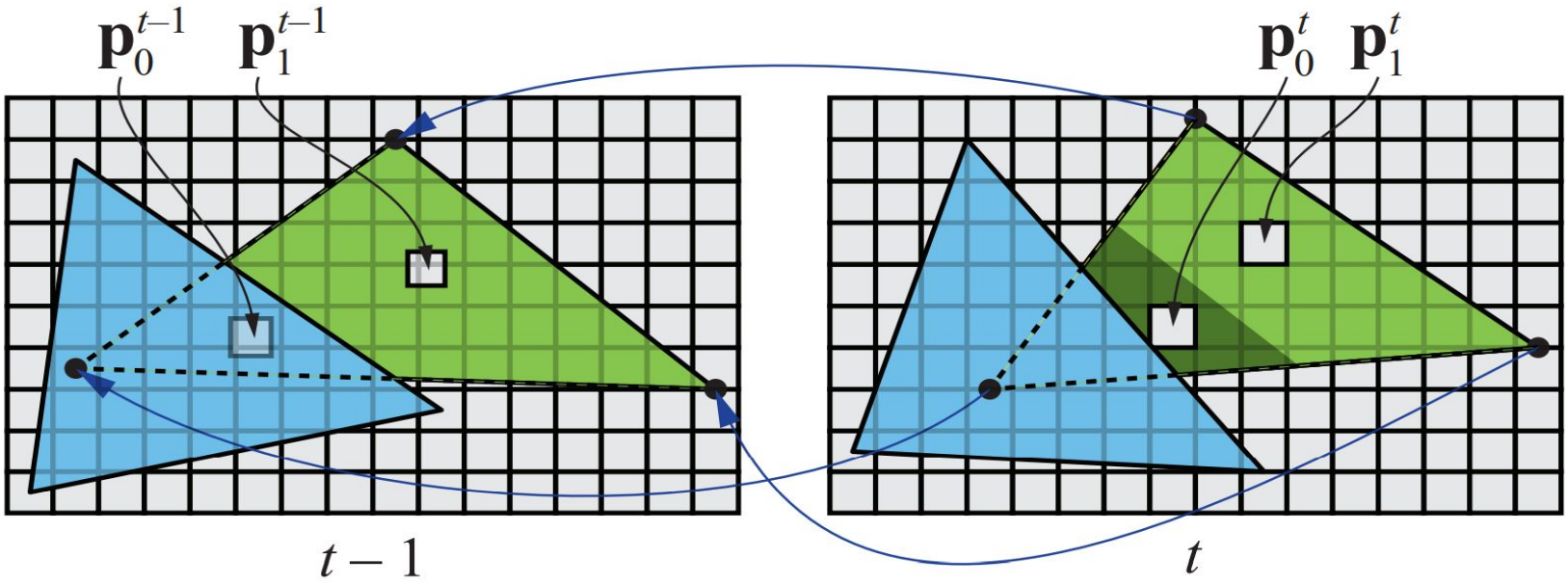
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

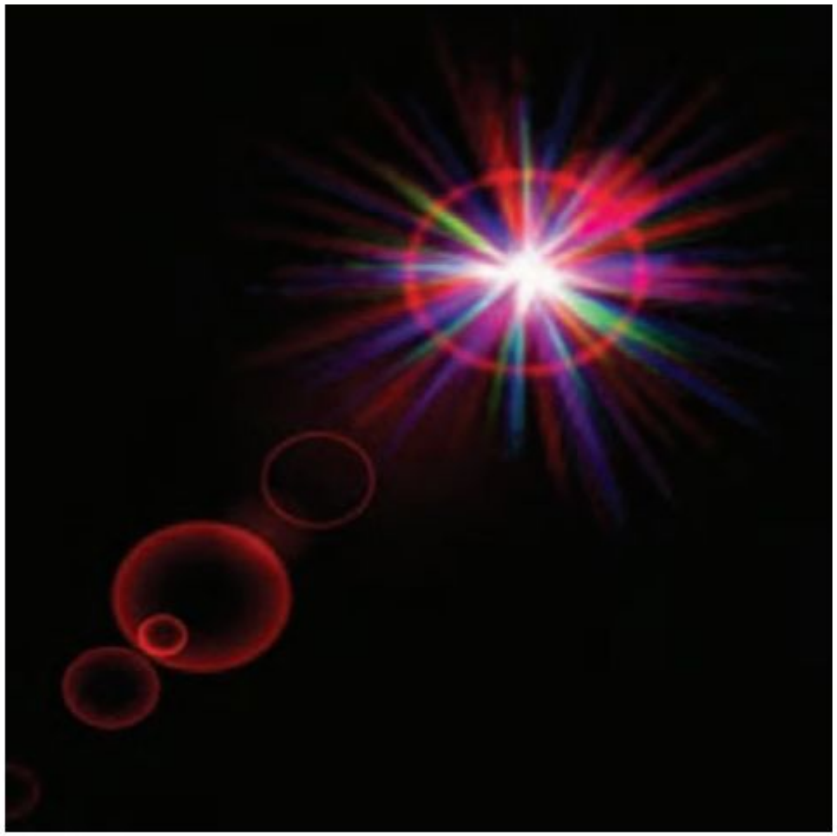
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

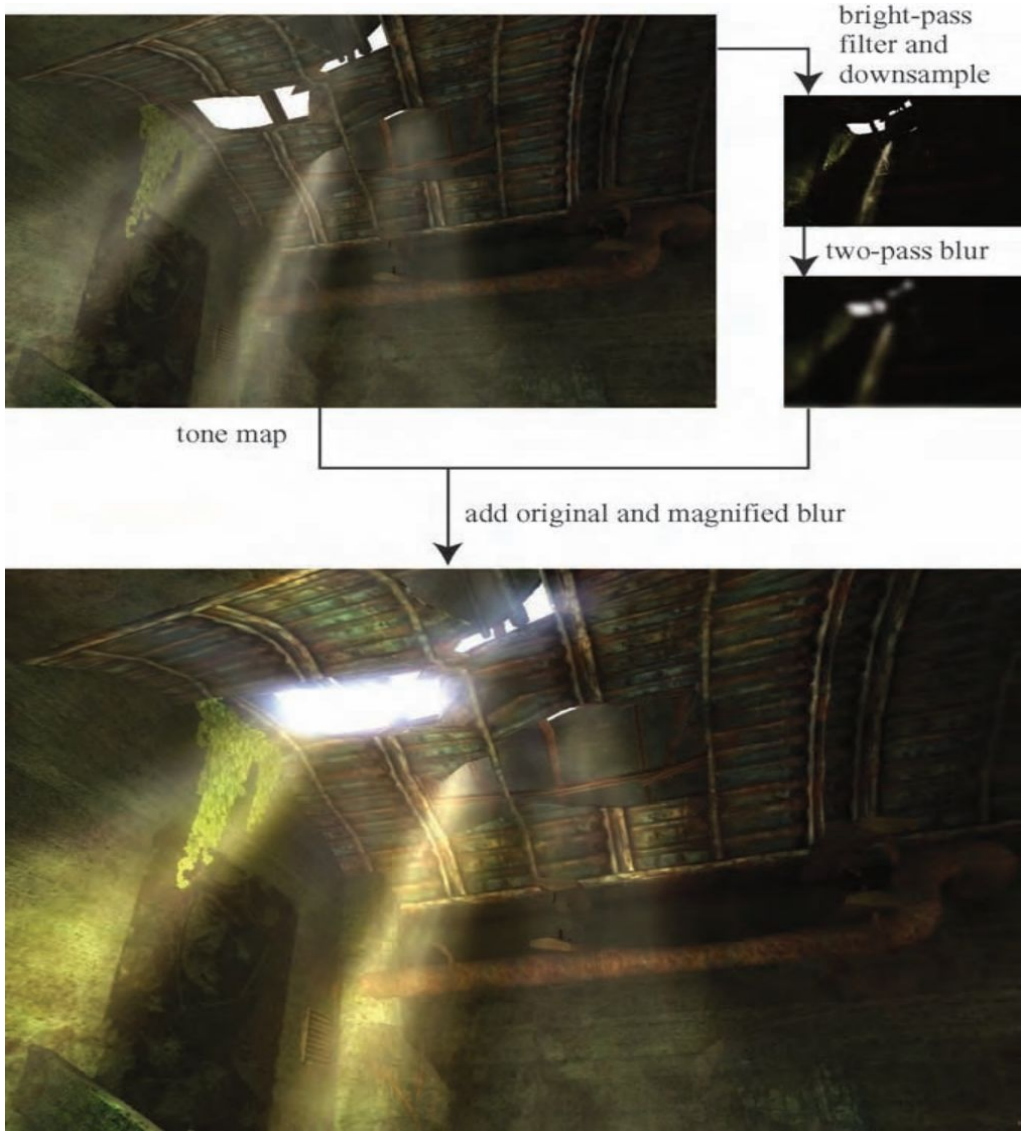
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

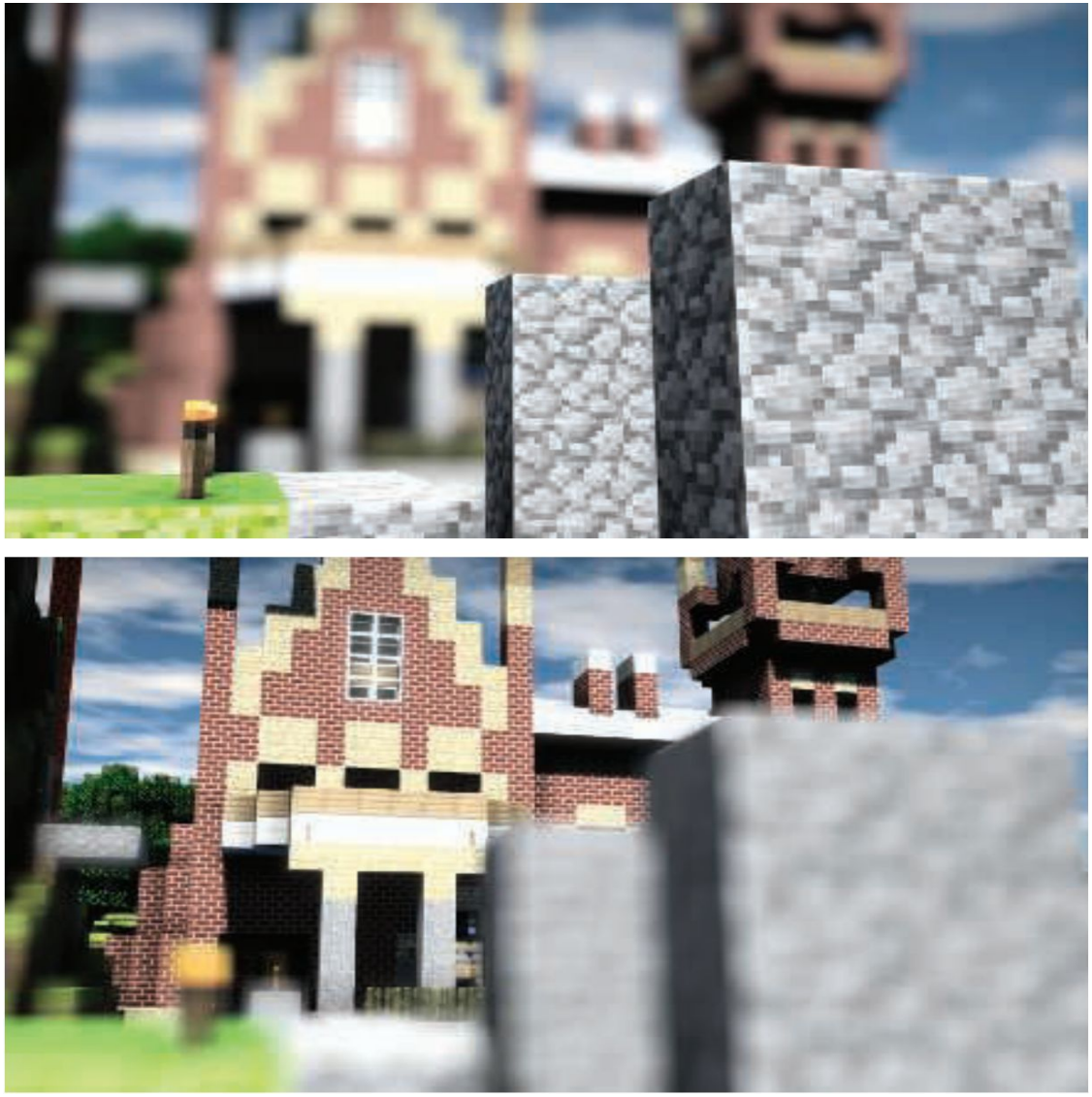
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

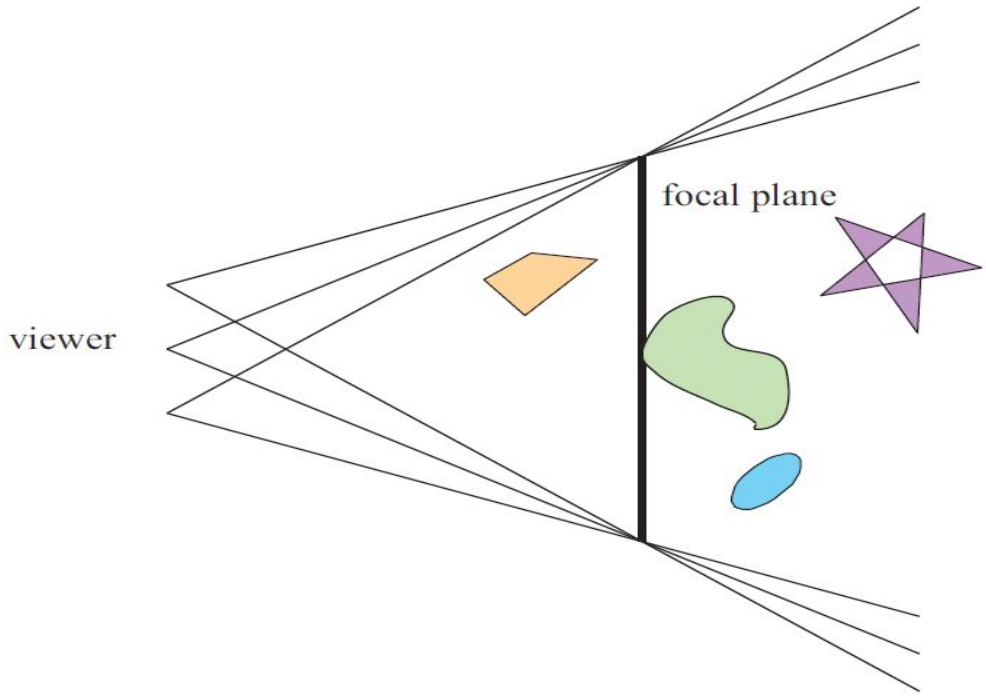


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

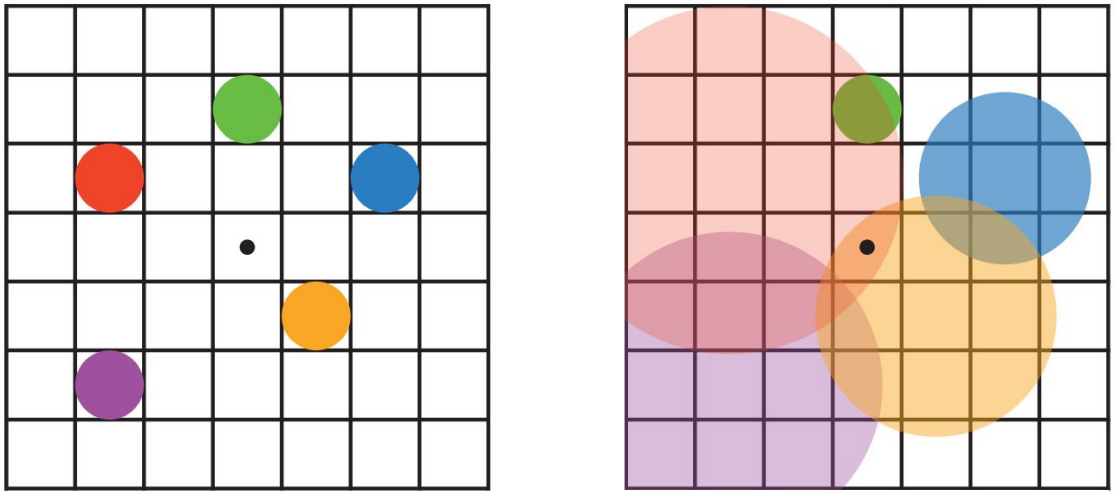
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

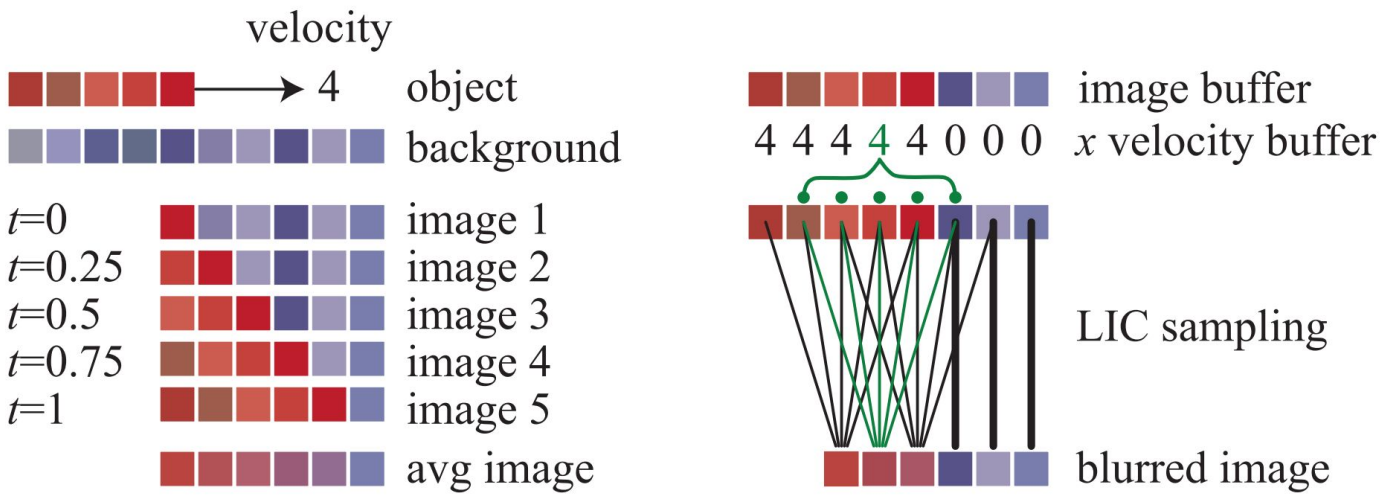
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

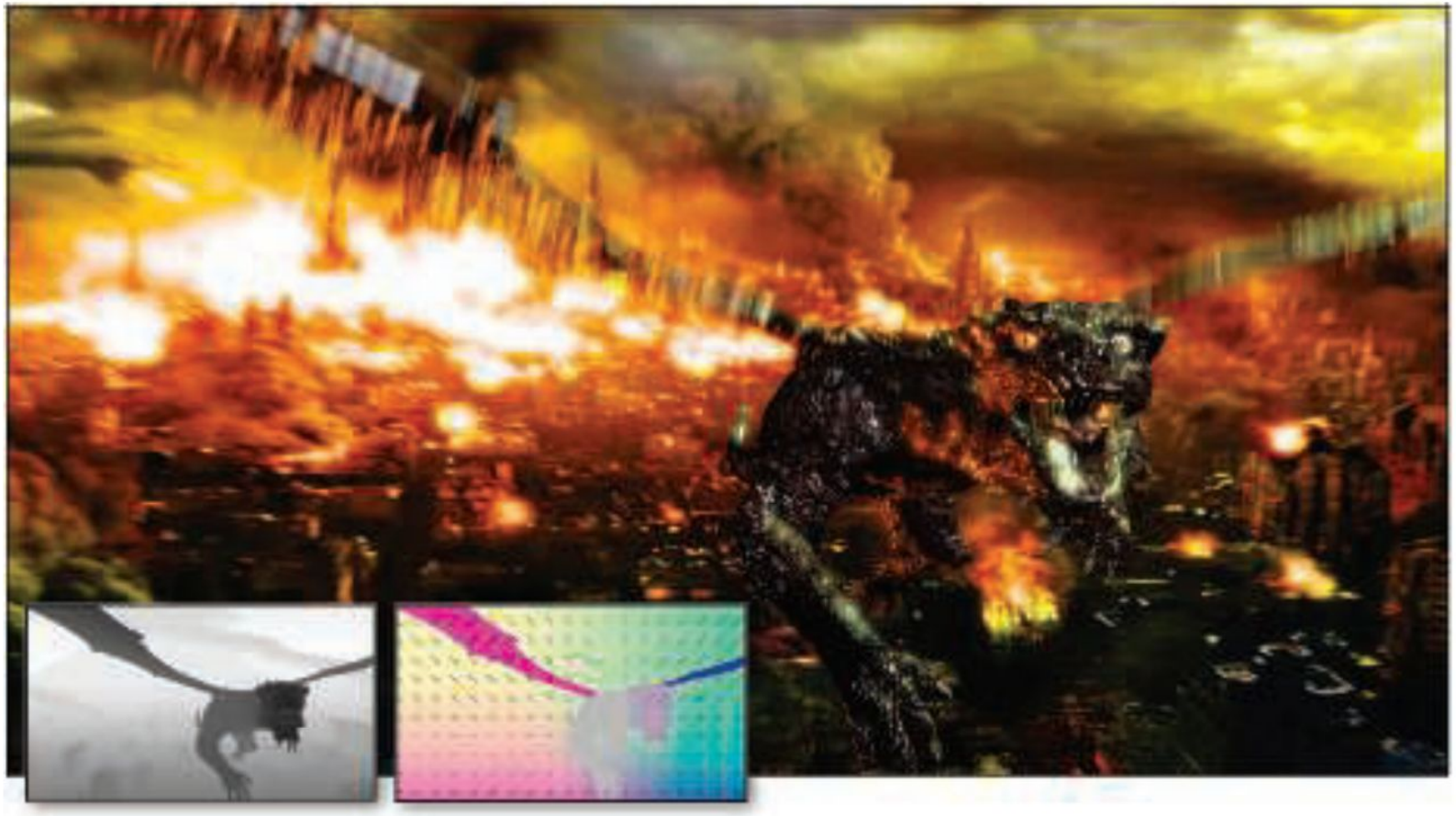
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

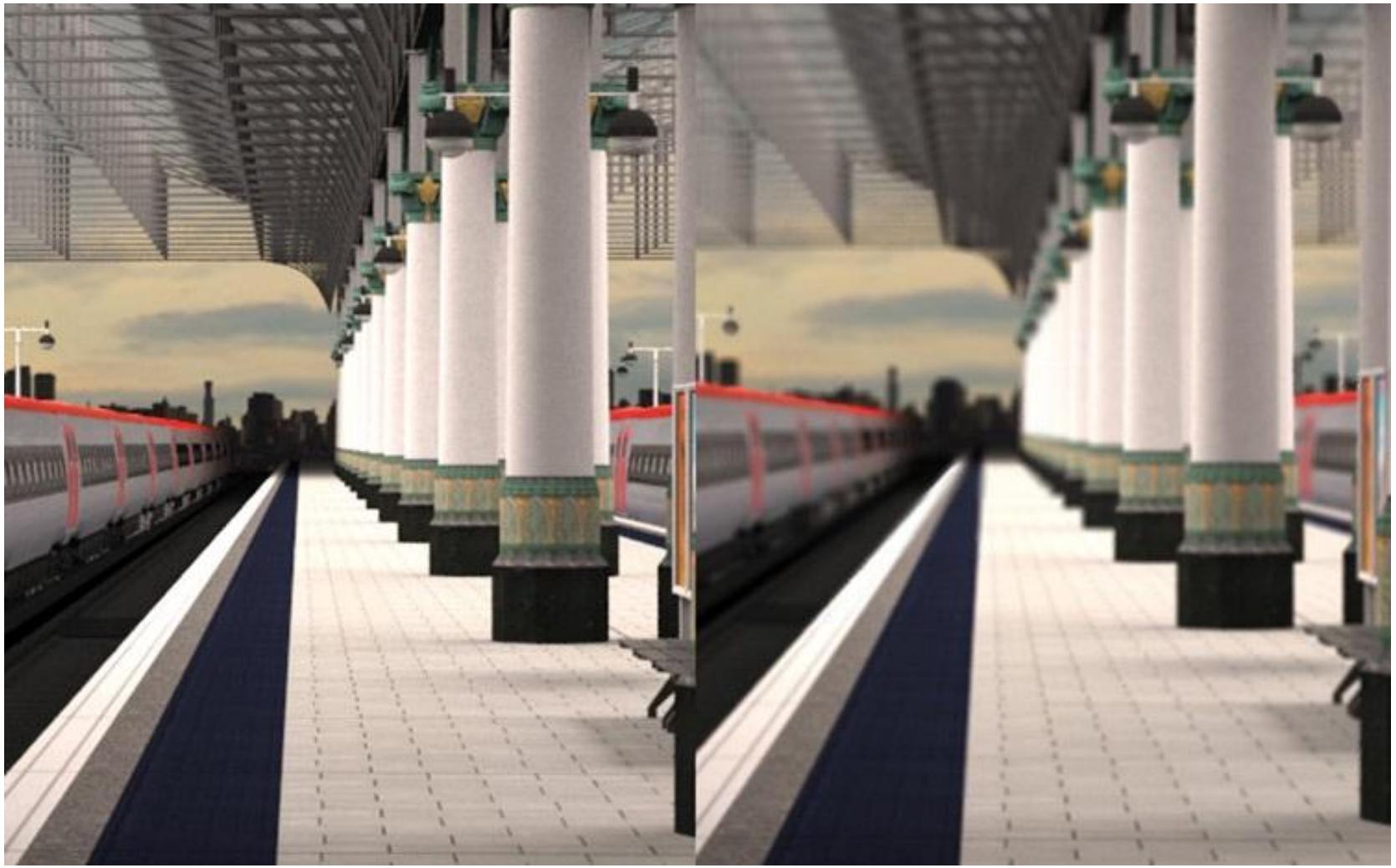
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

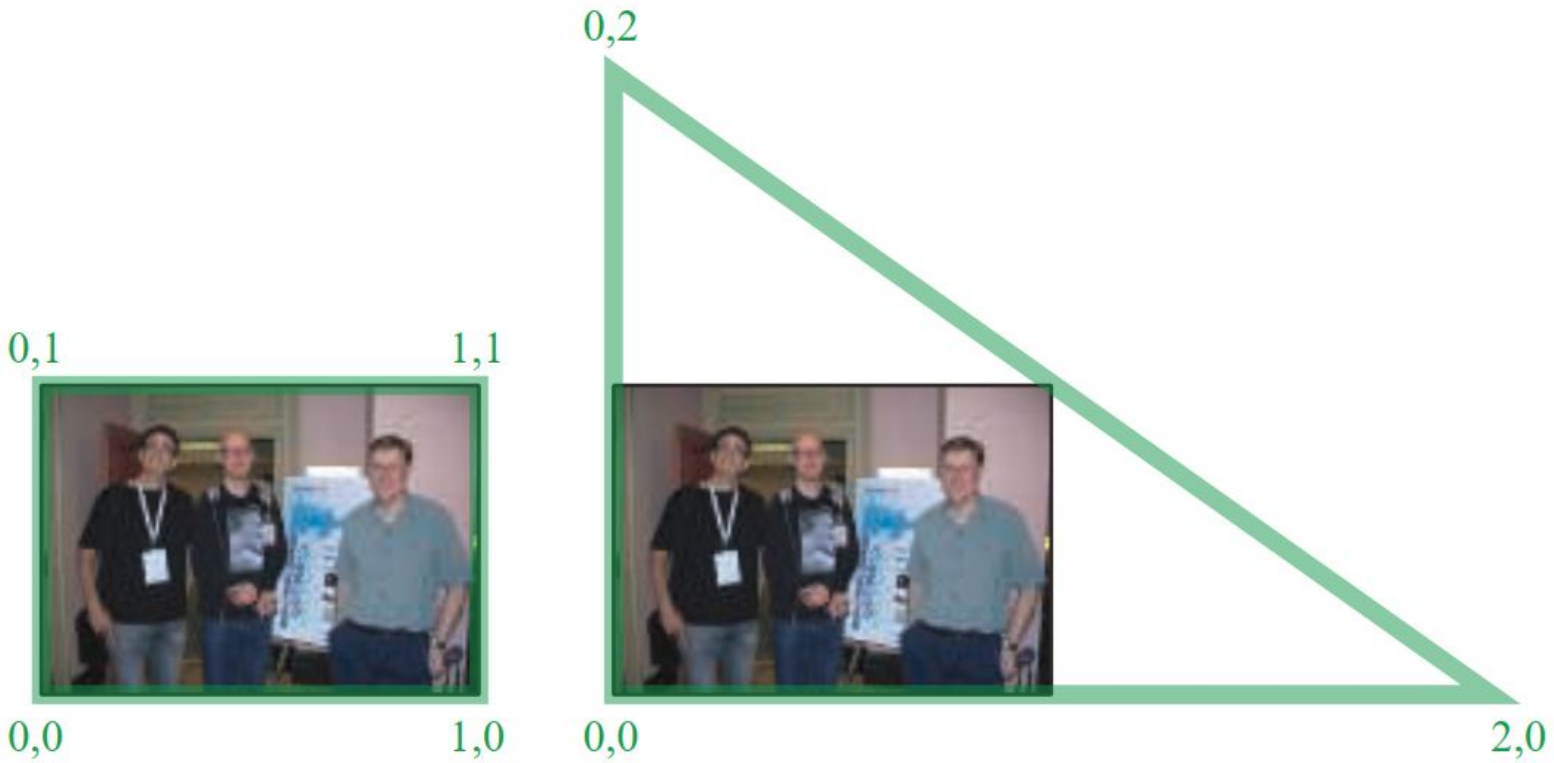
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.
Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

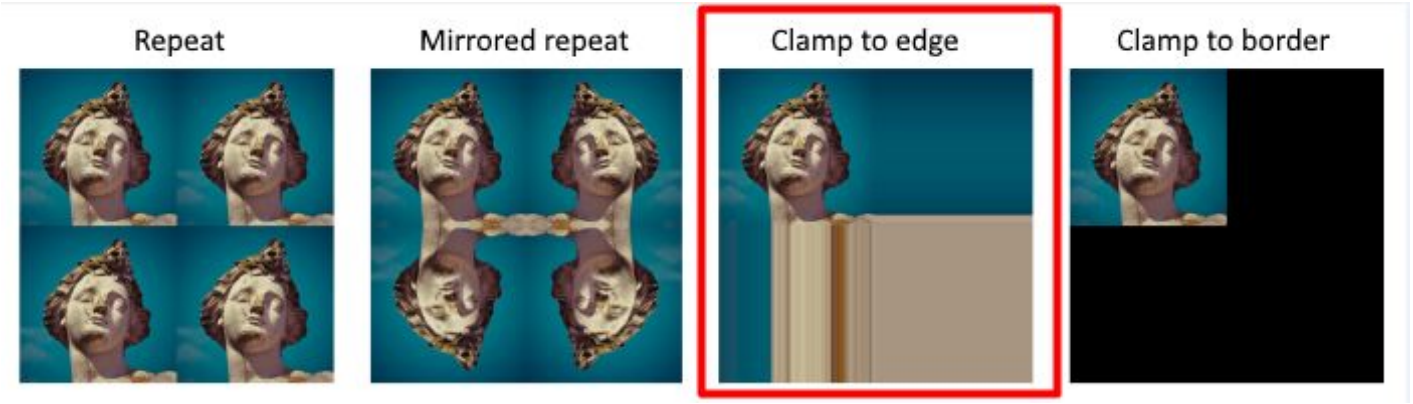
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

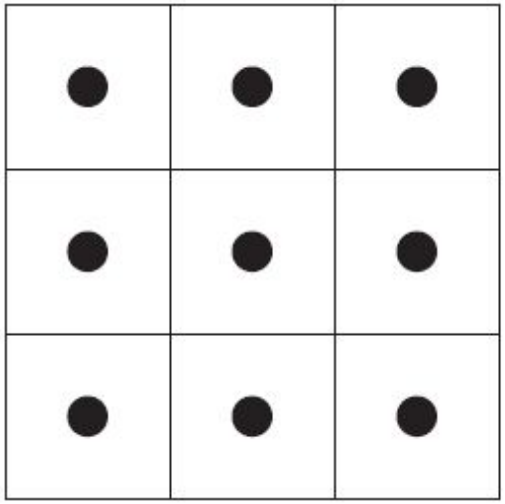
(c)

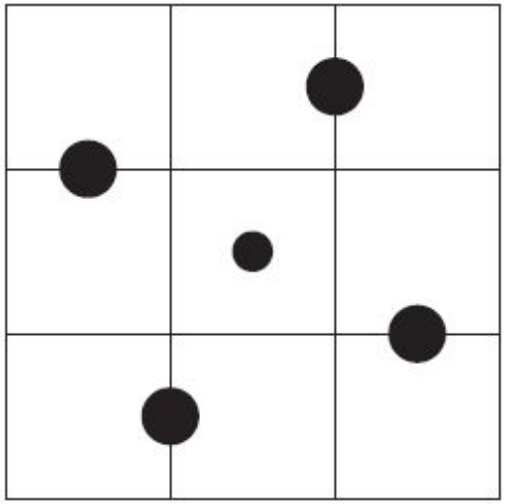
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

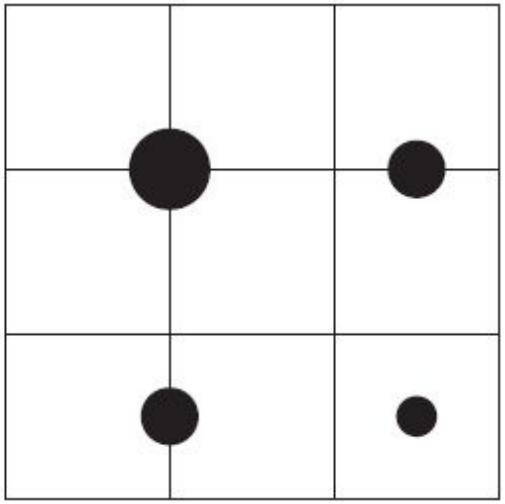
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

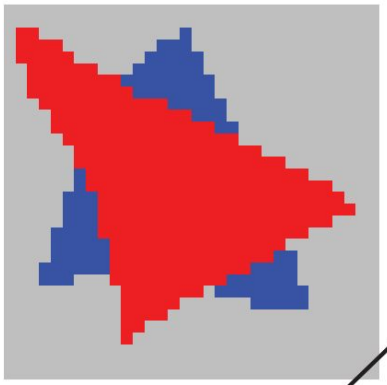
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

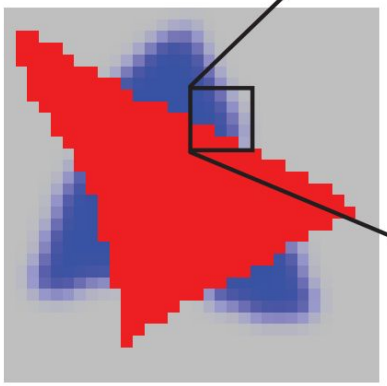
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

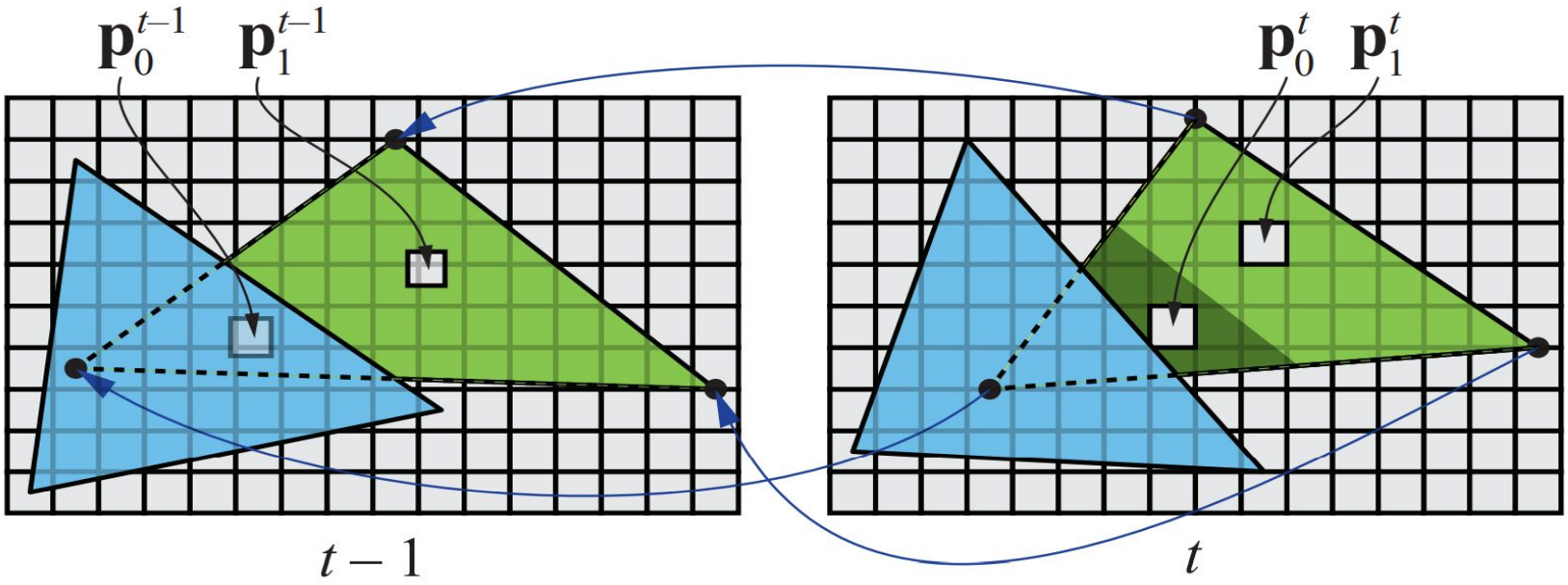
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

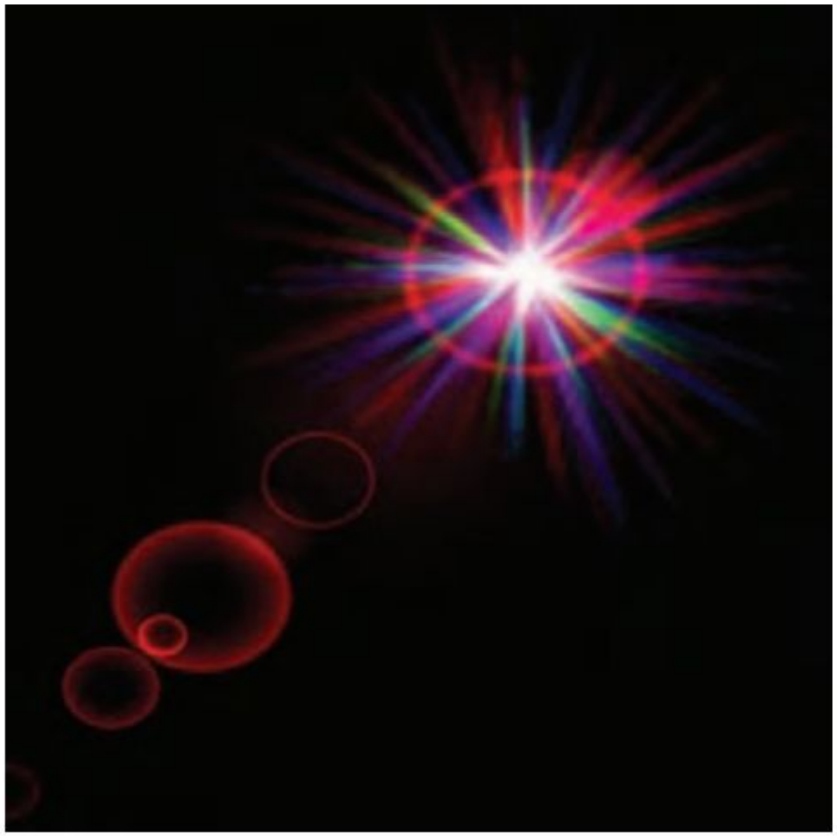
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

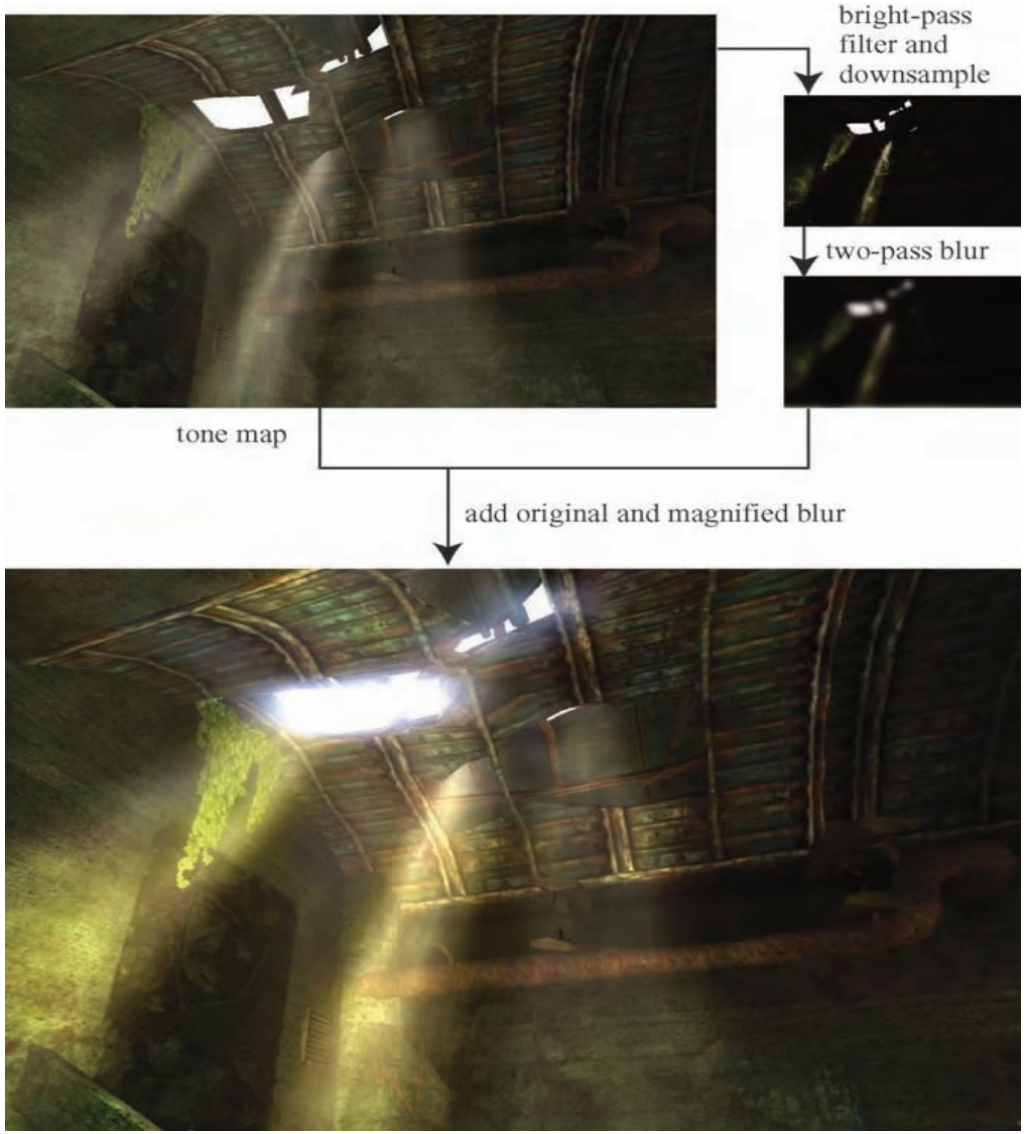
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

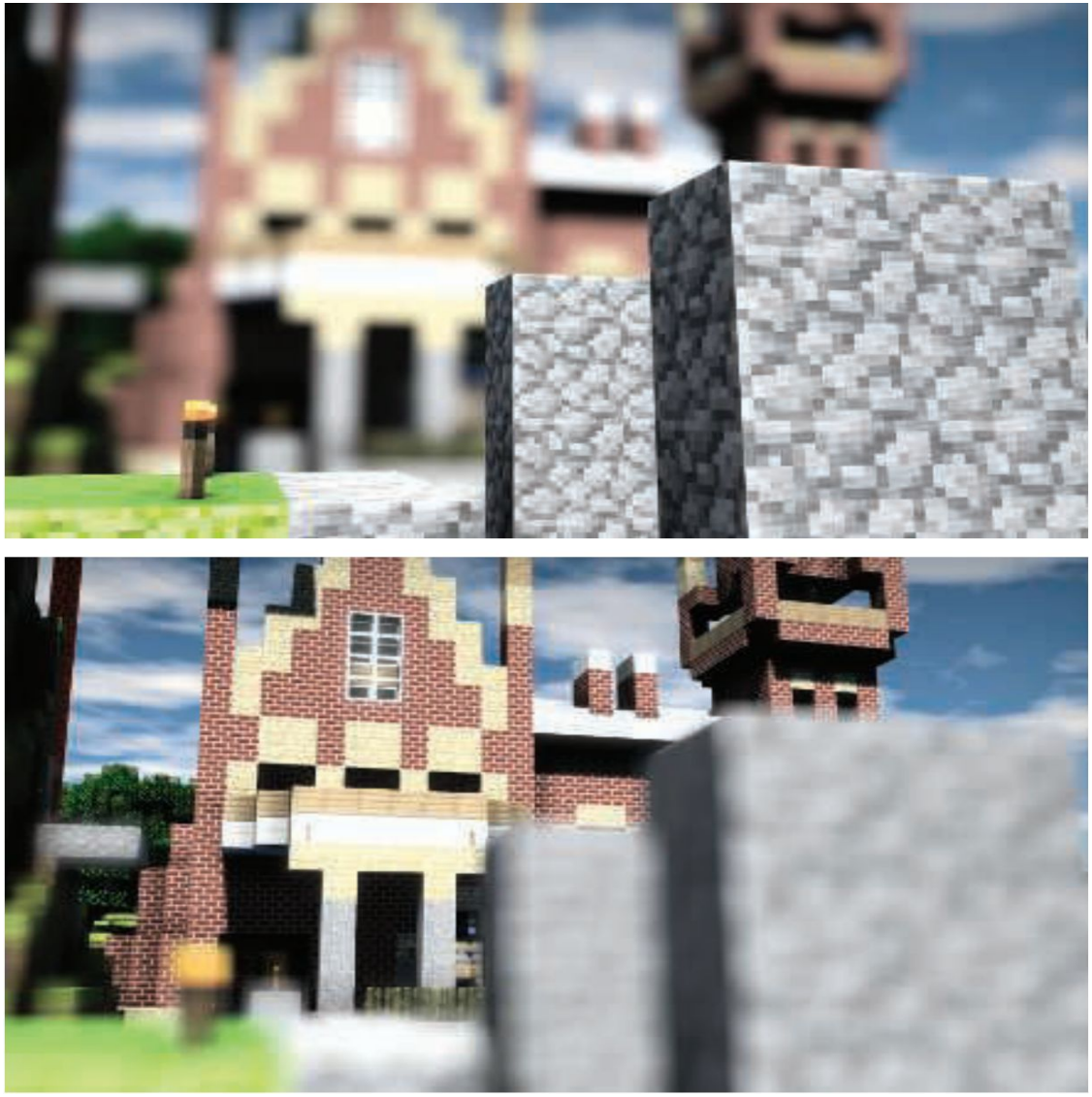
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

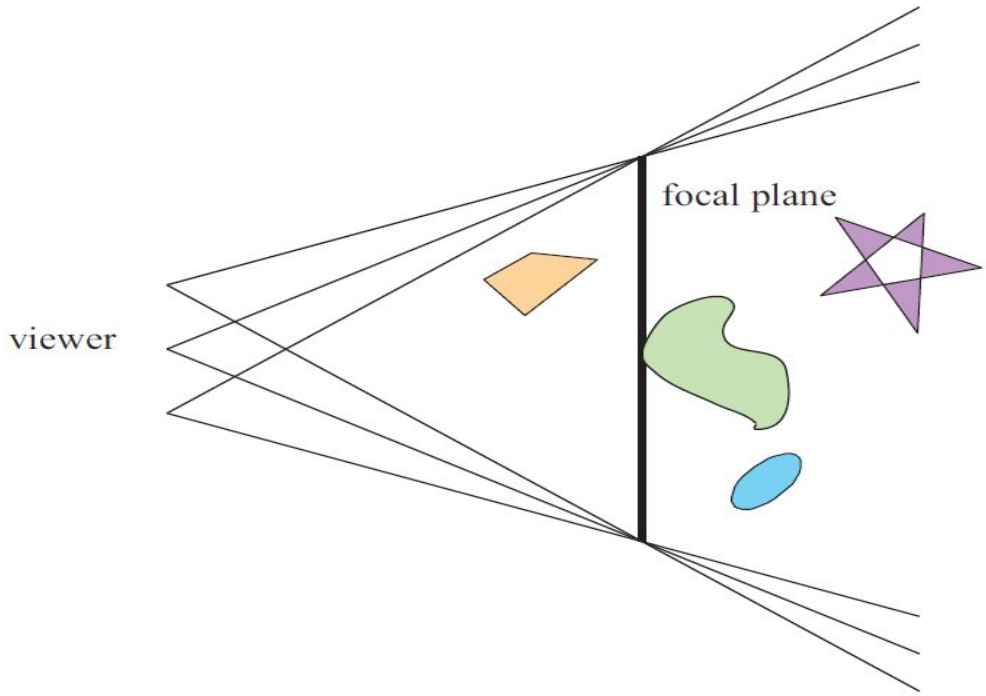


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

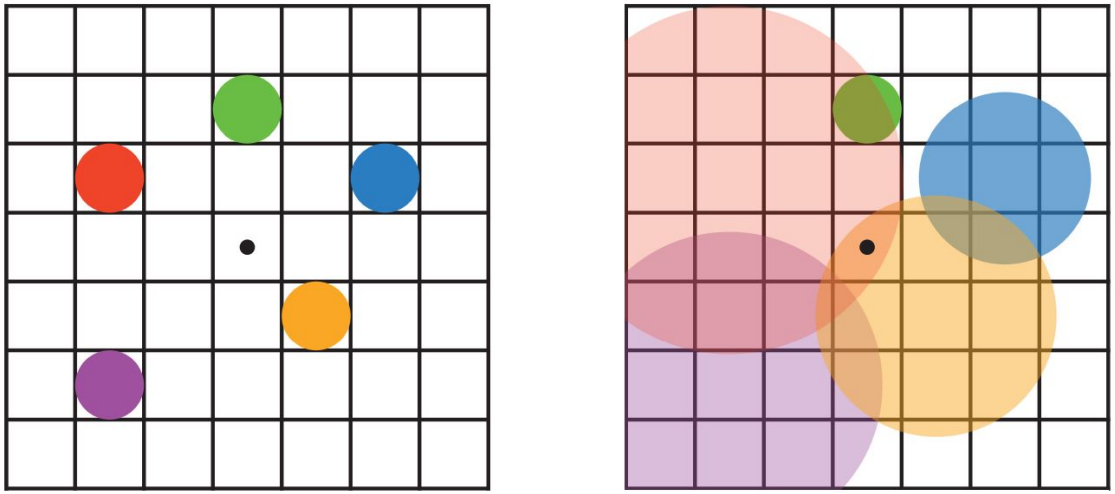
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

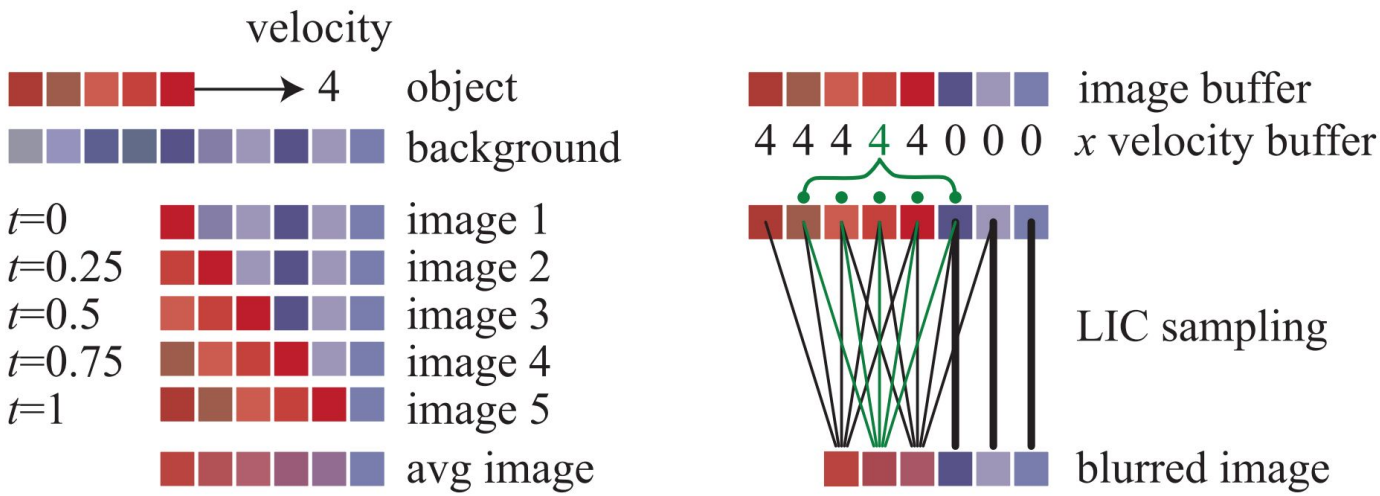
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

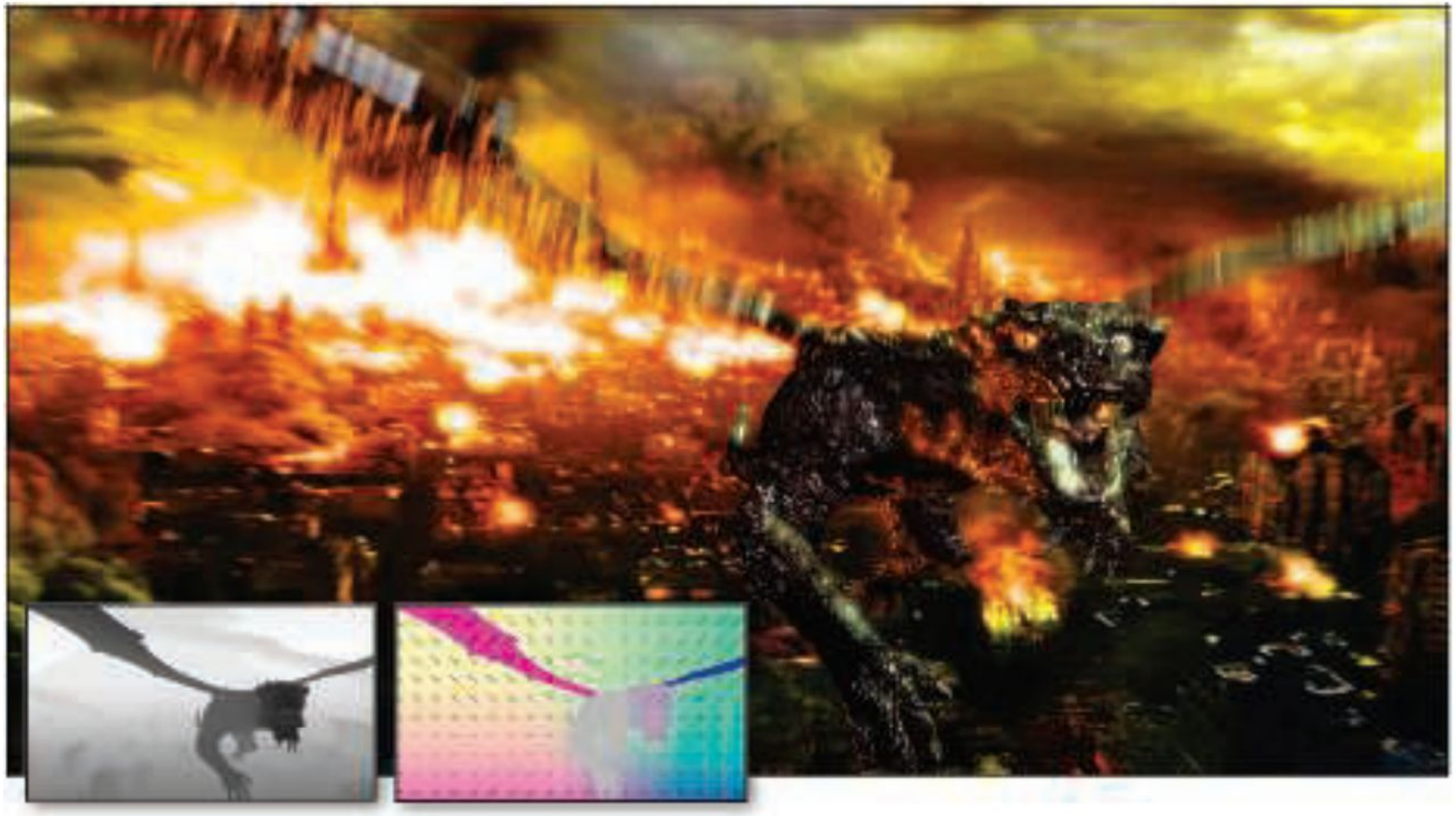
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

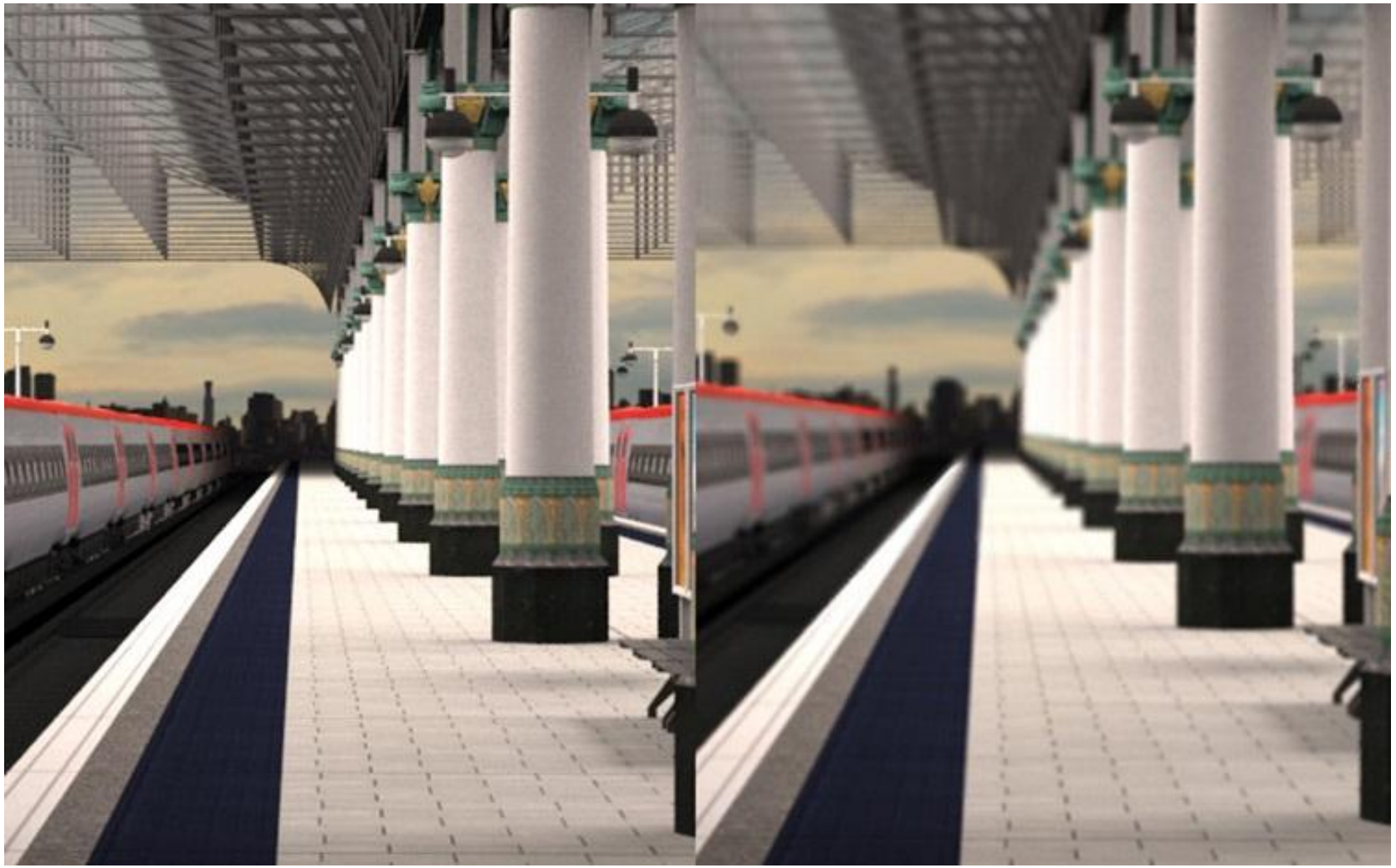
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

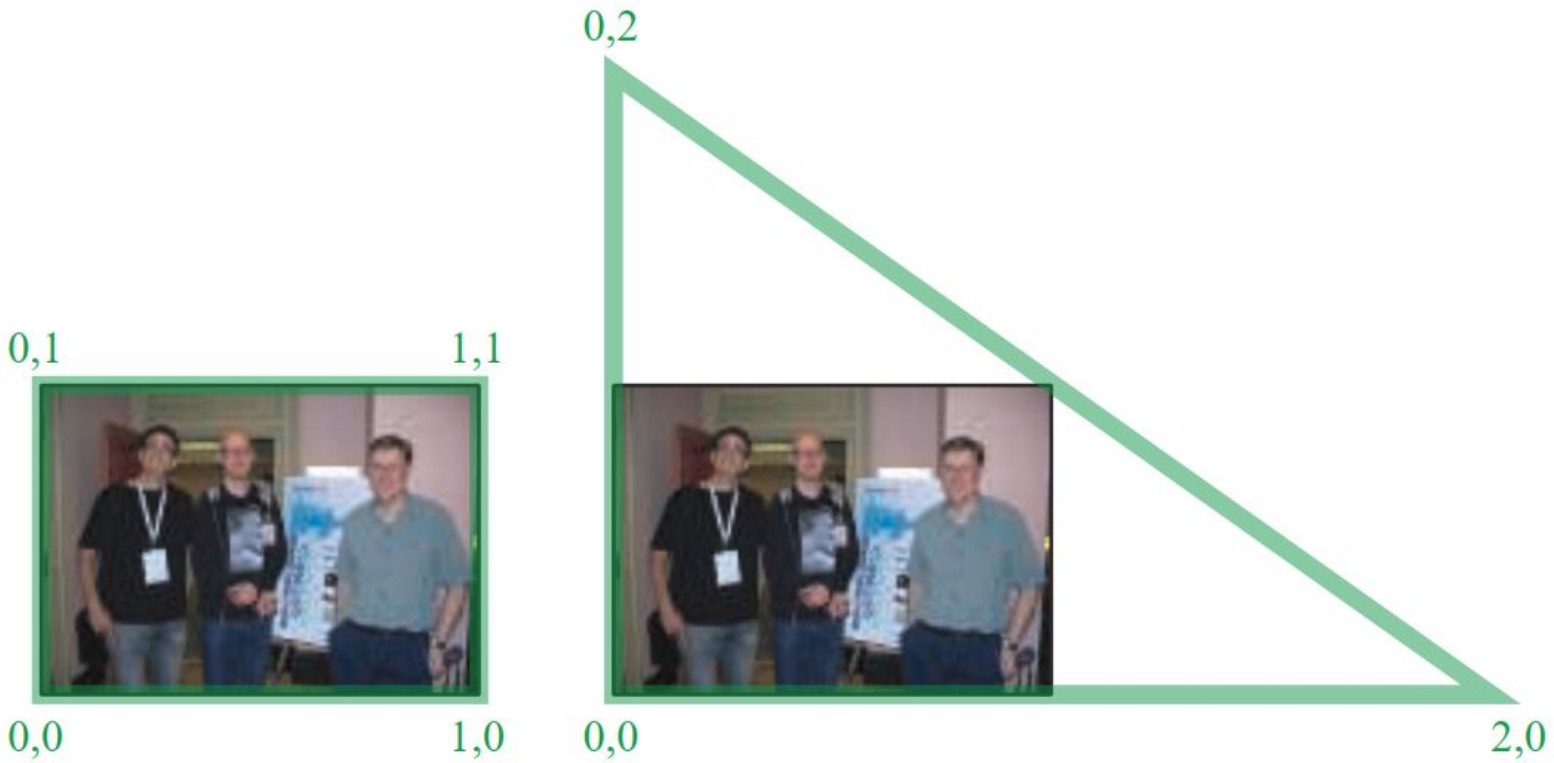
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.

Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

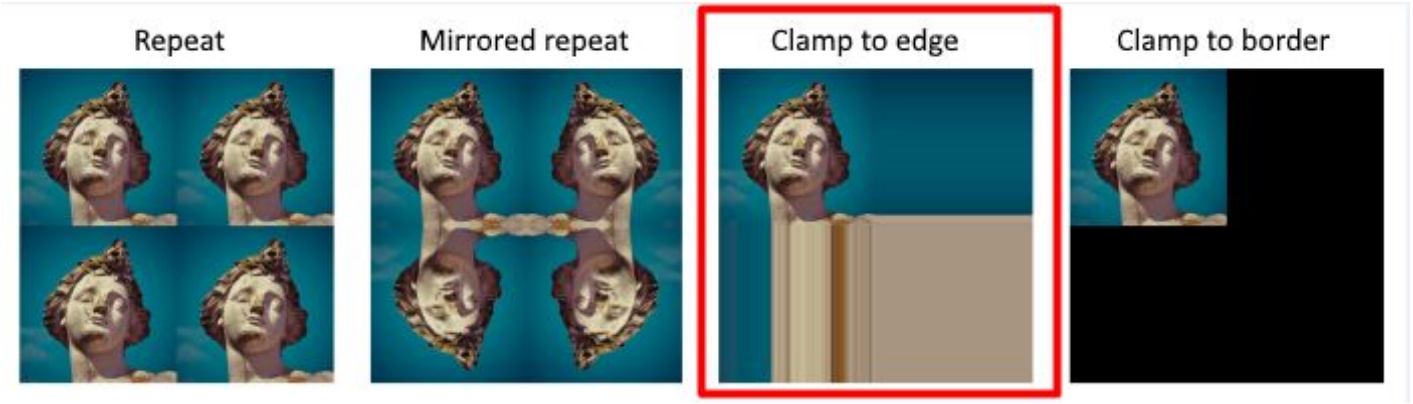
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

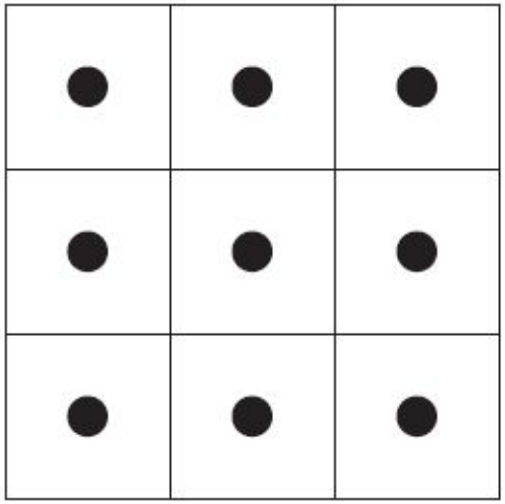
(c)

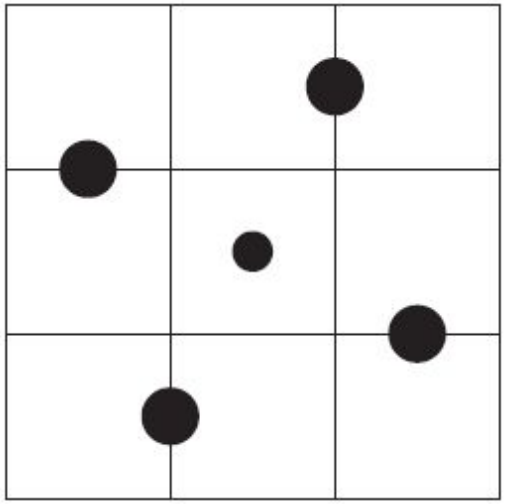
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

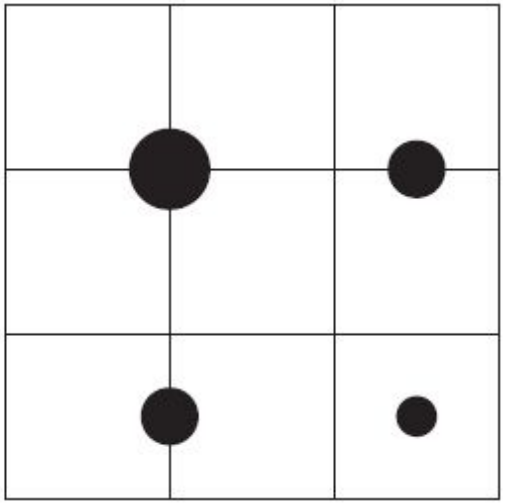
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

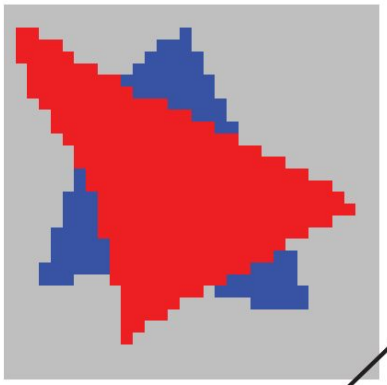
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

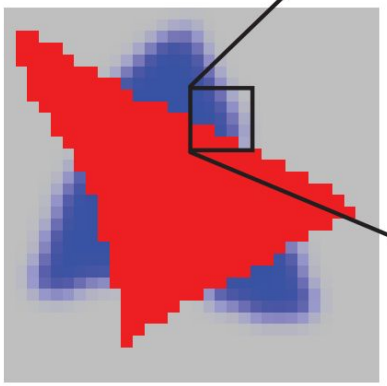
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

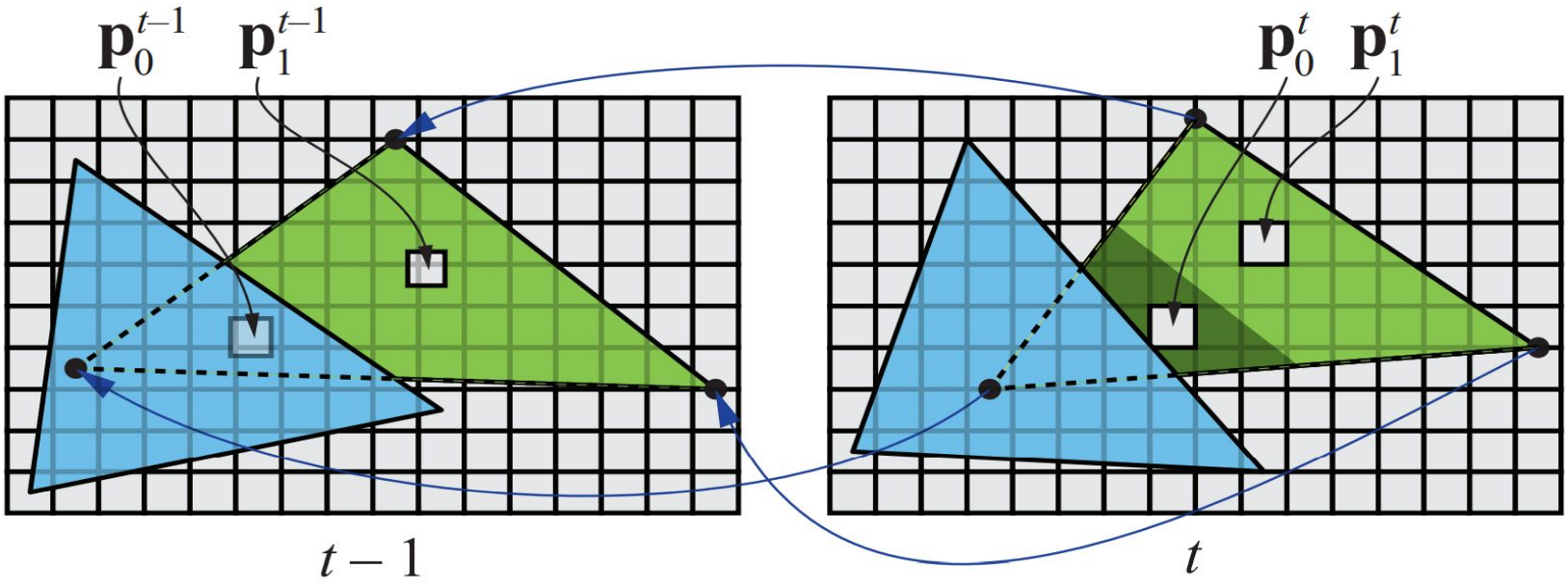
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

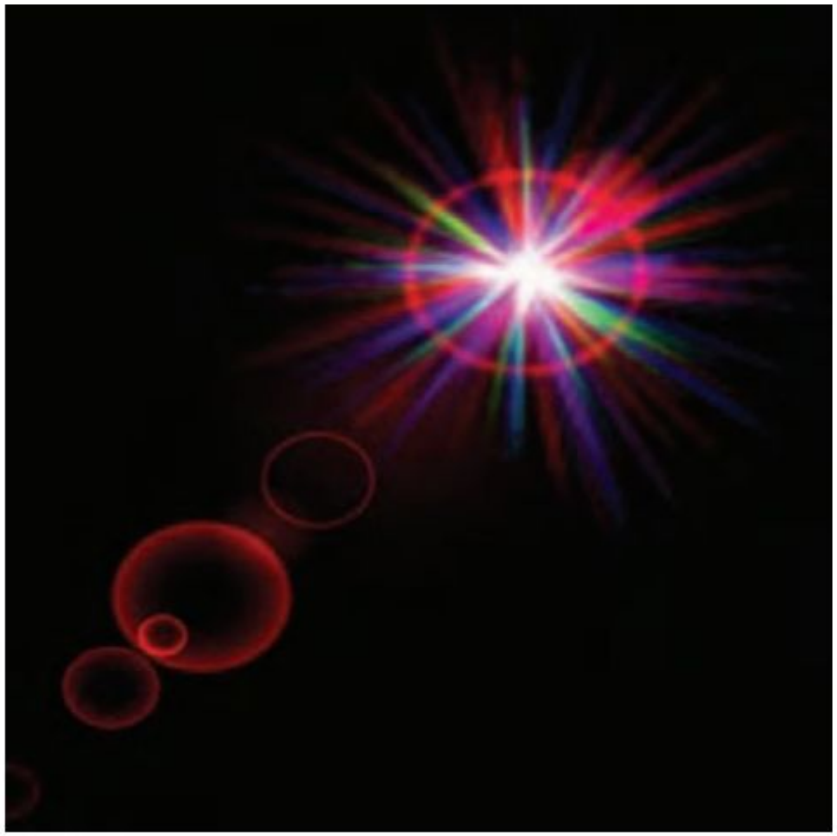
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

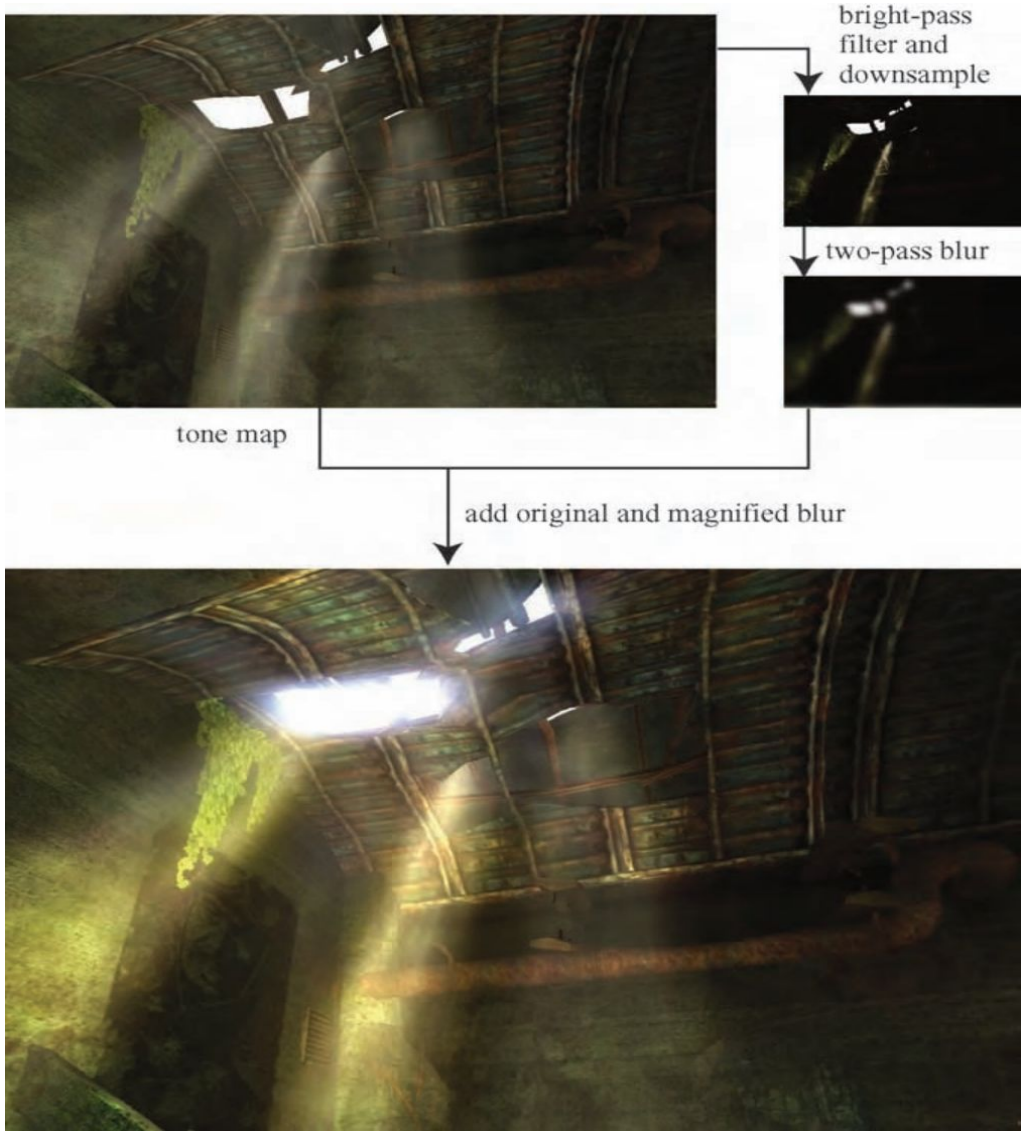
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

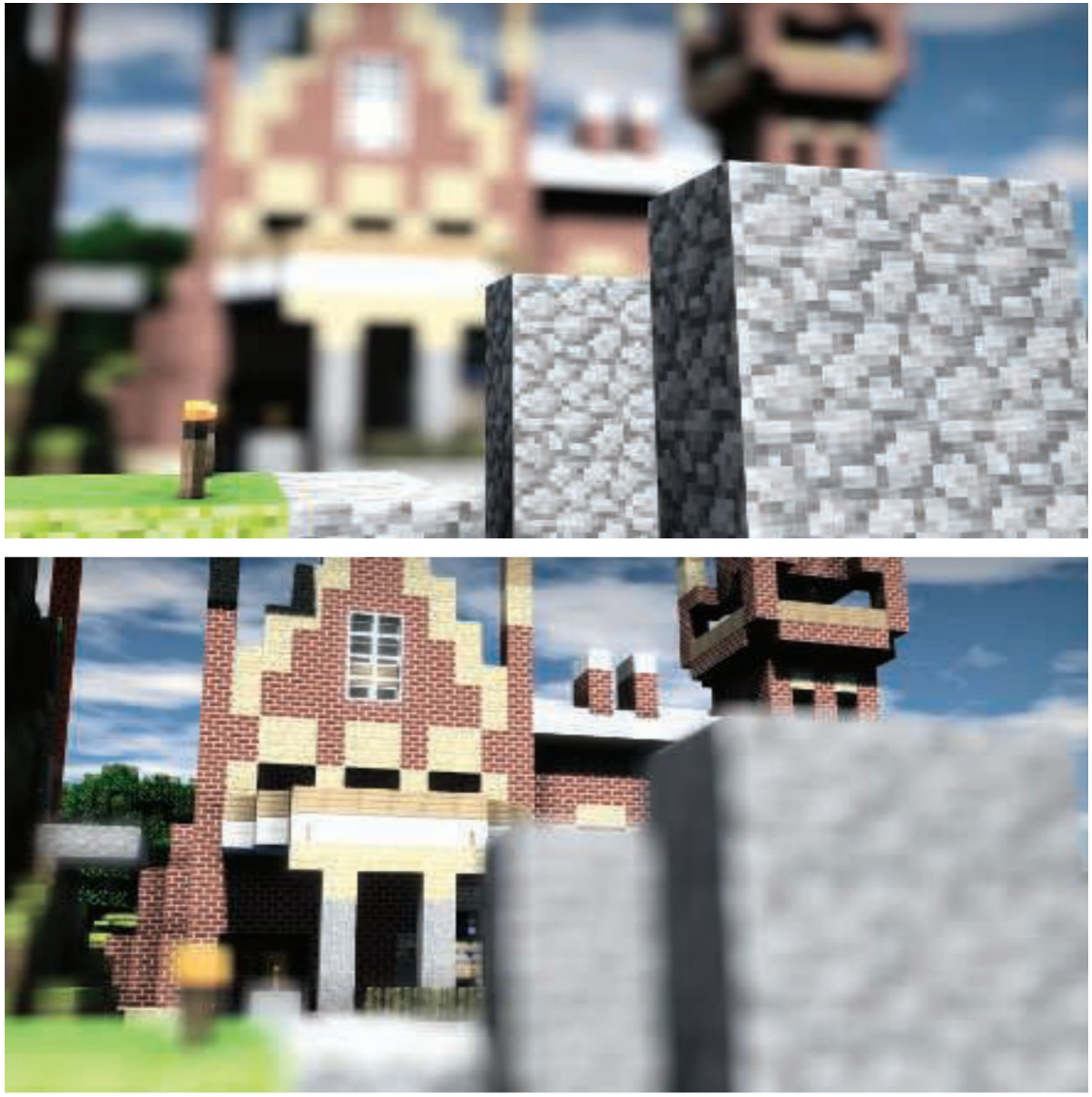
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

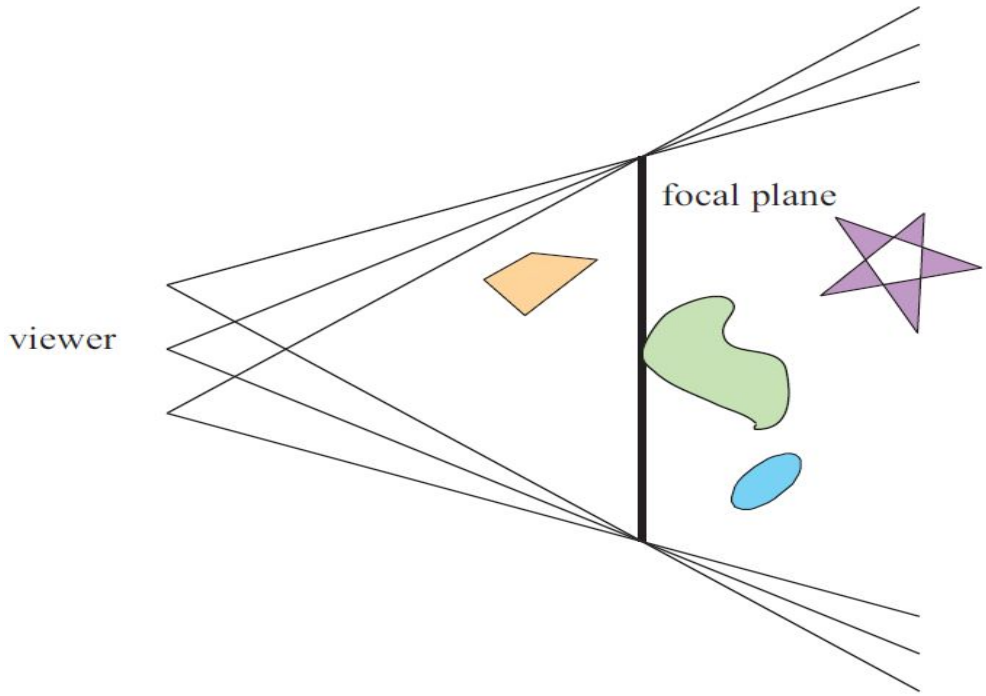


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

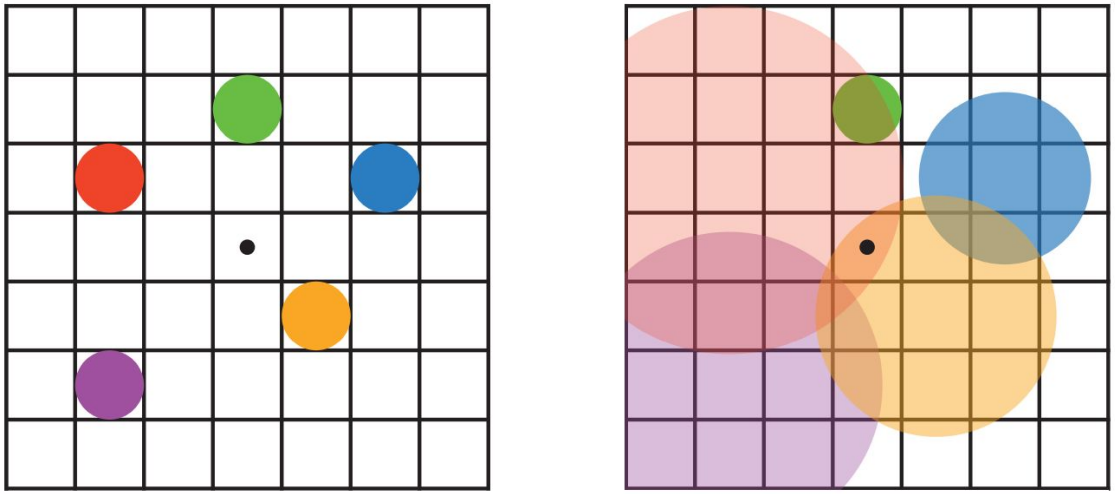
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

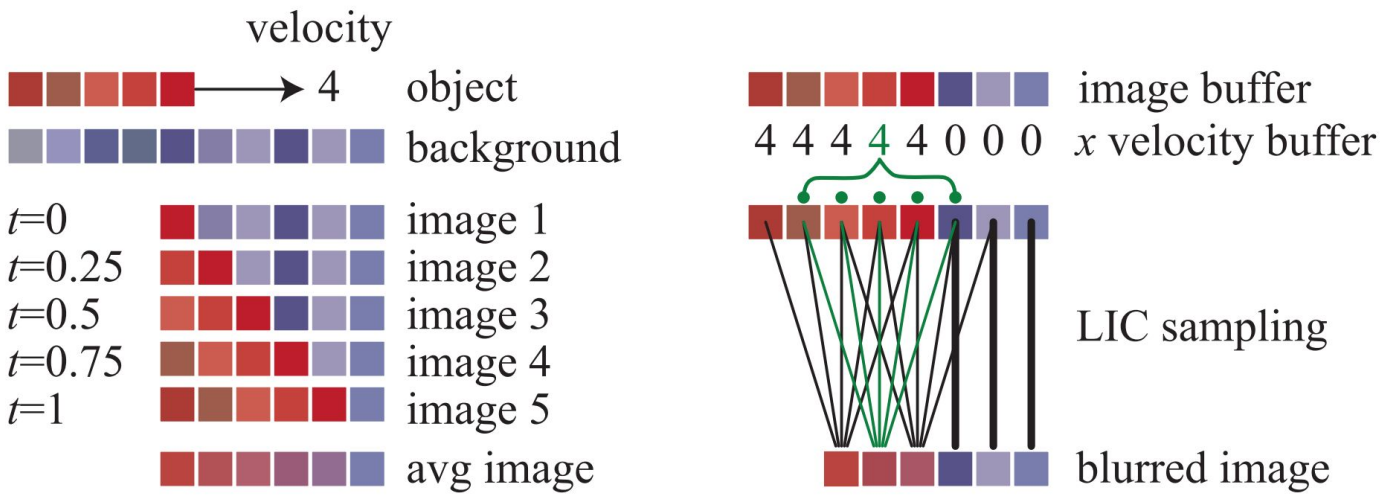
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

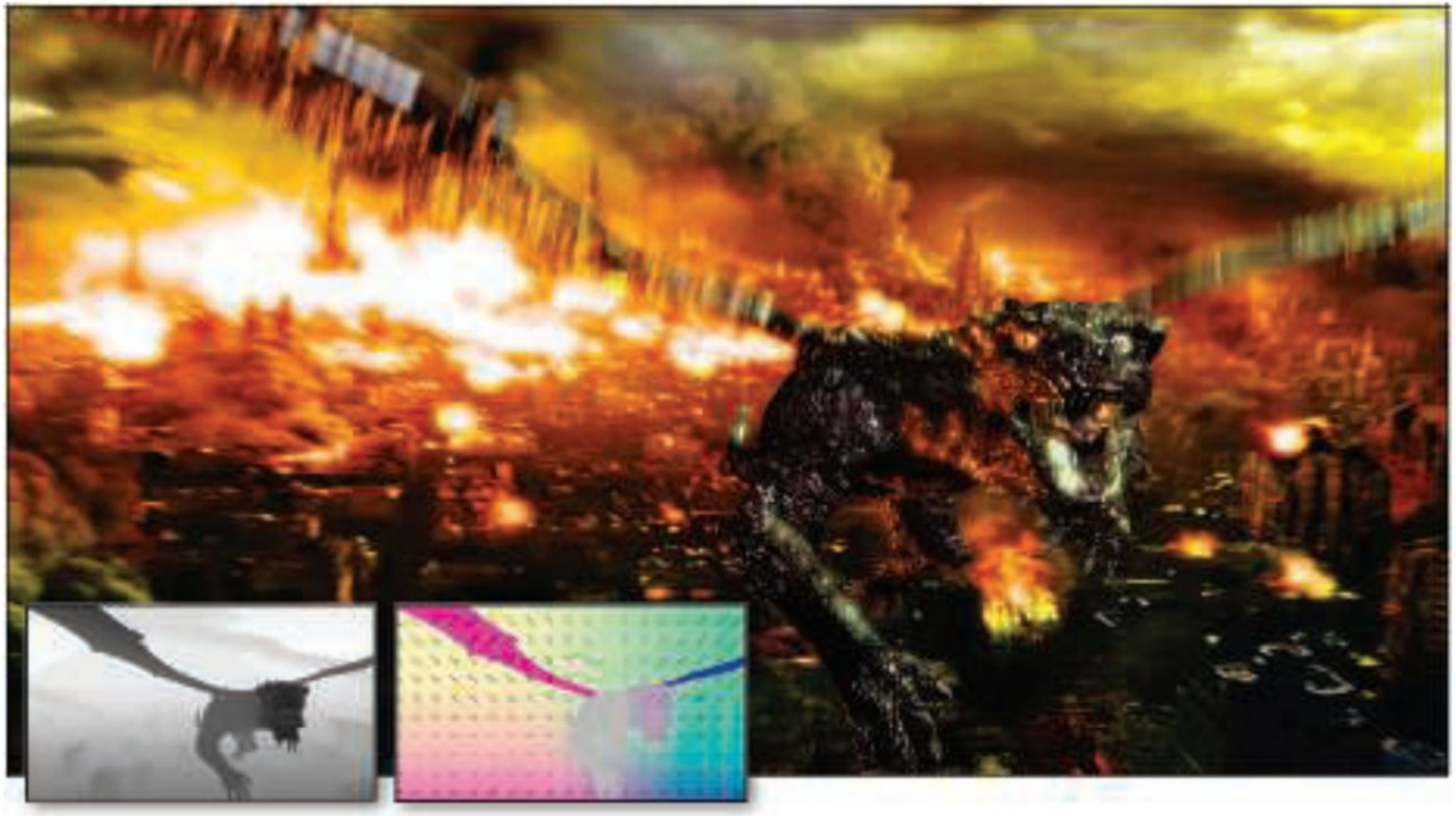
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

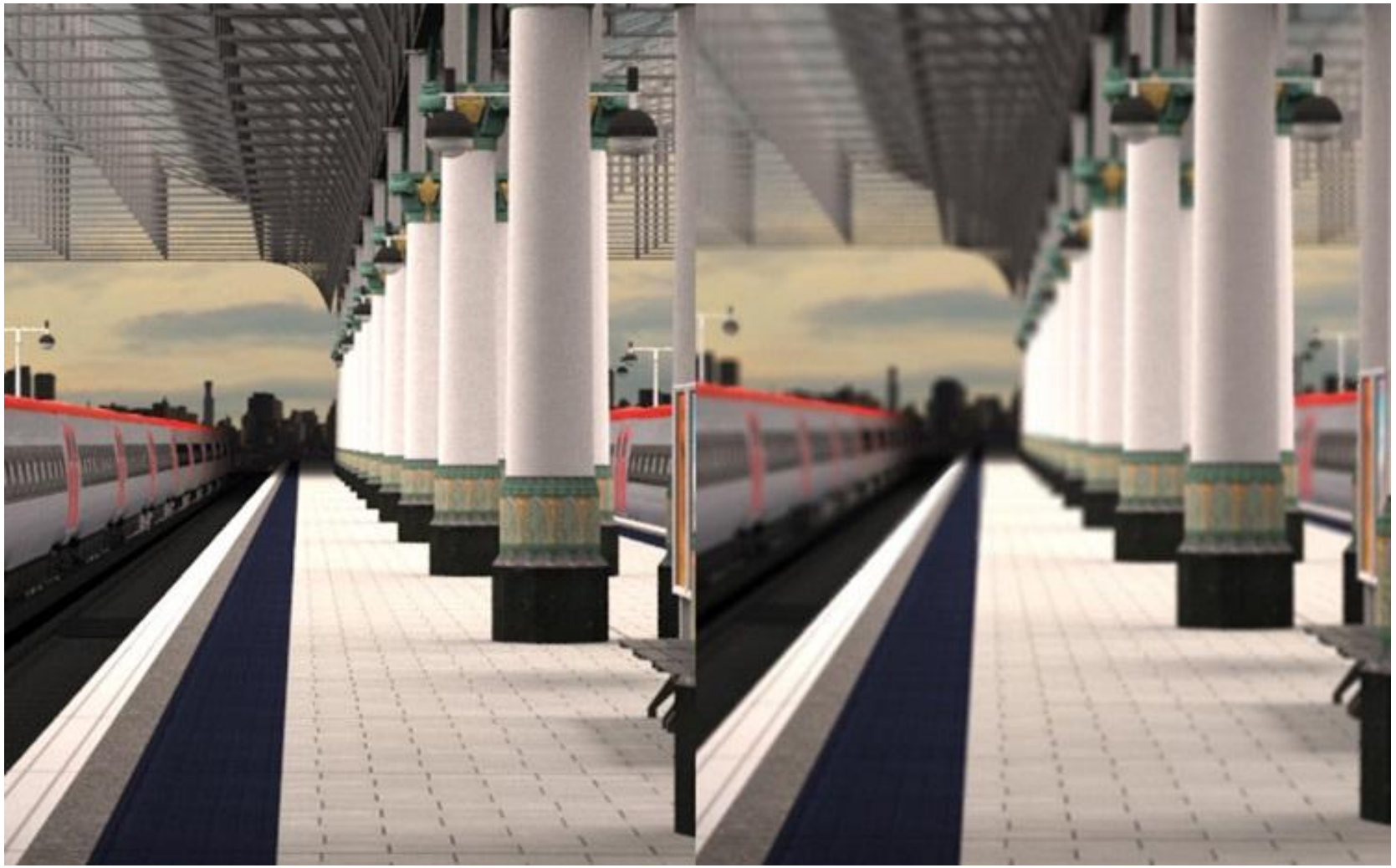
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

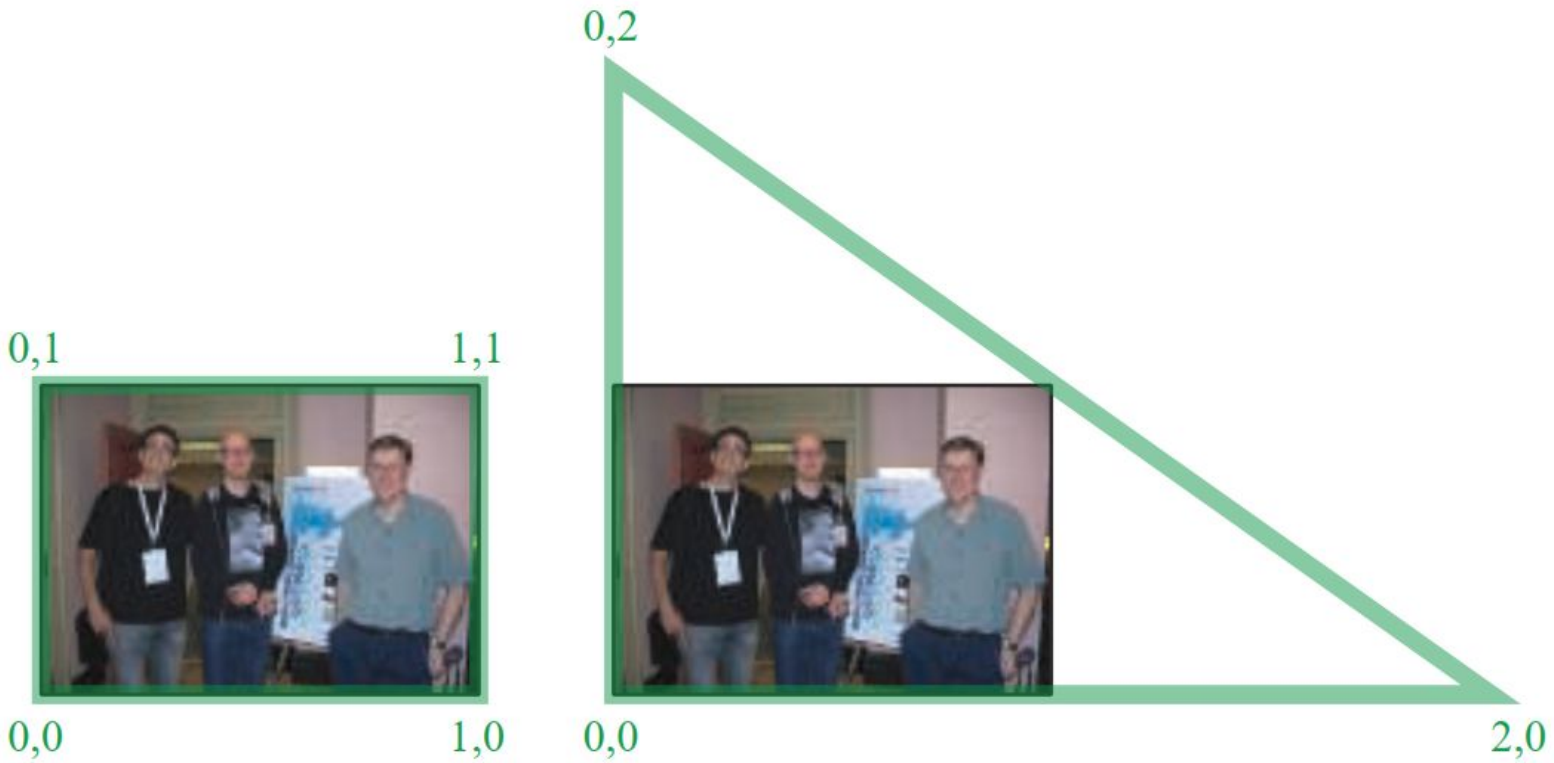
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.

Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

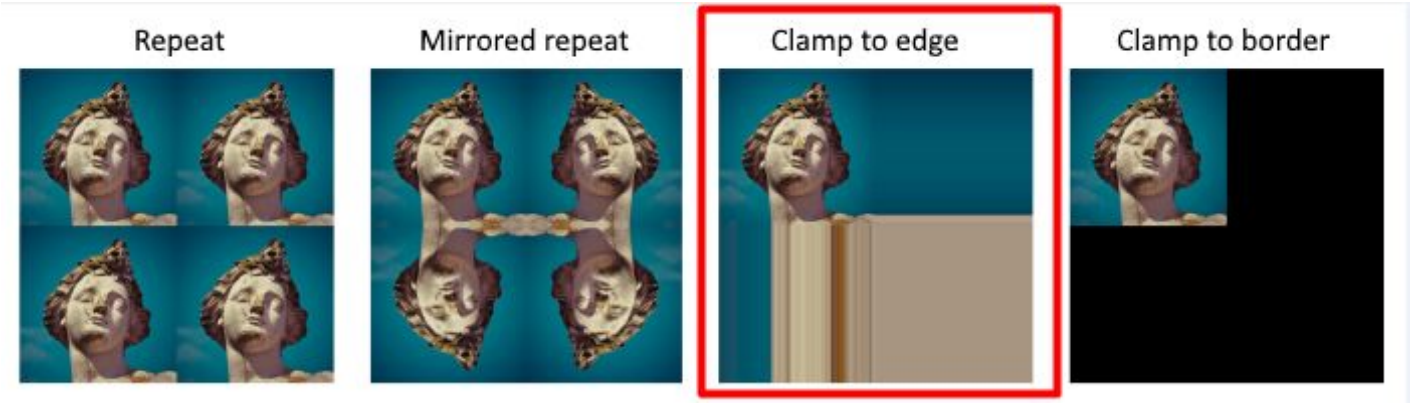
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

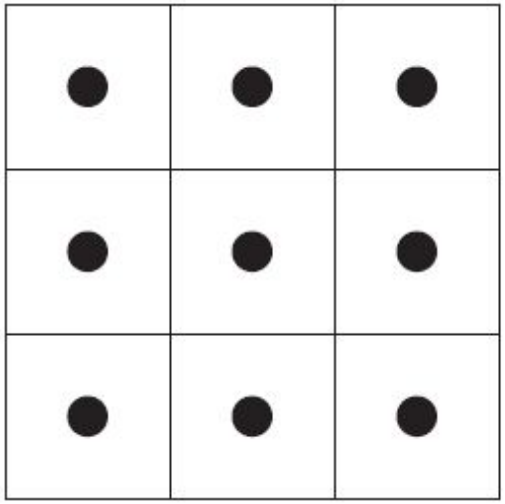
(c)

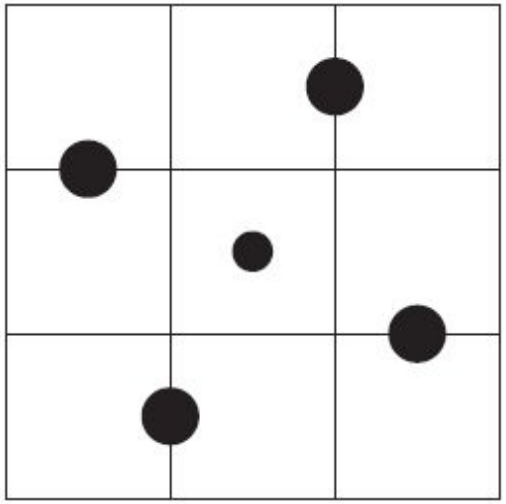
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

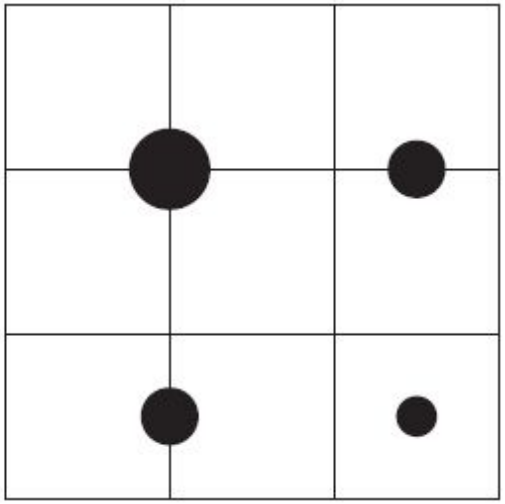
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

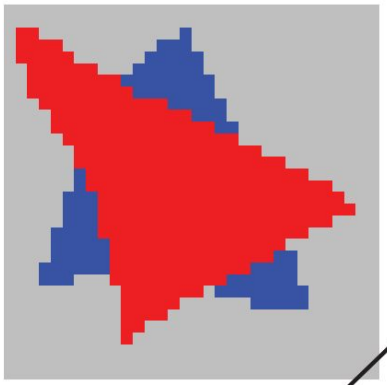
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

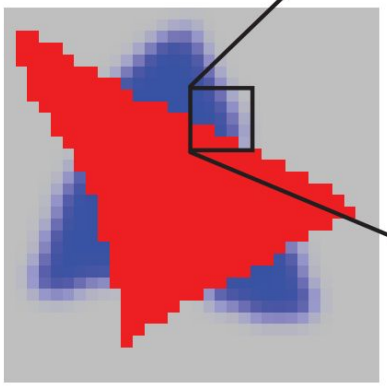
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

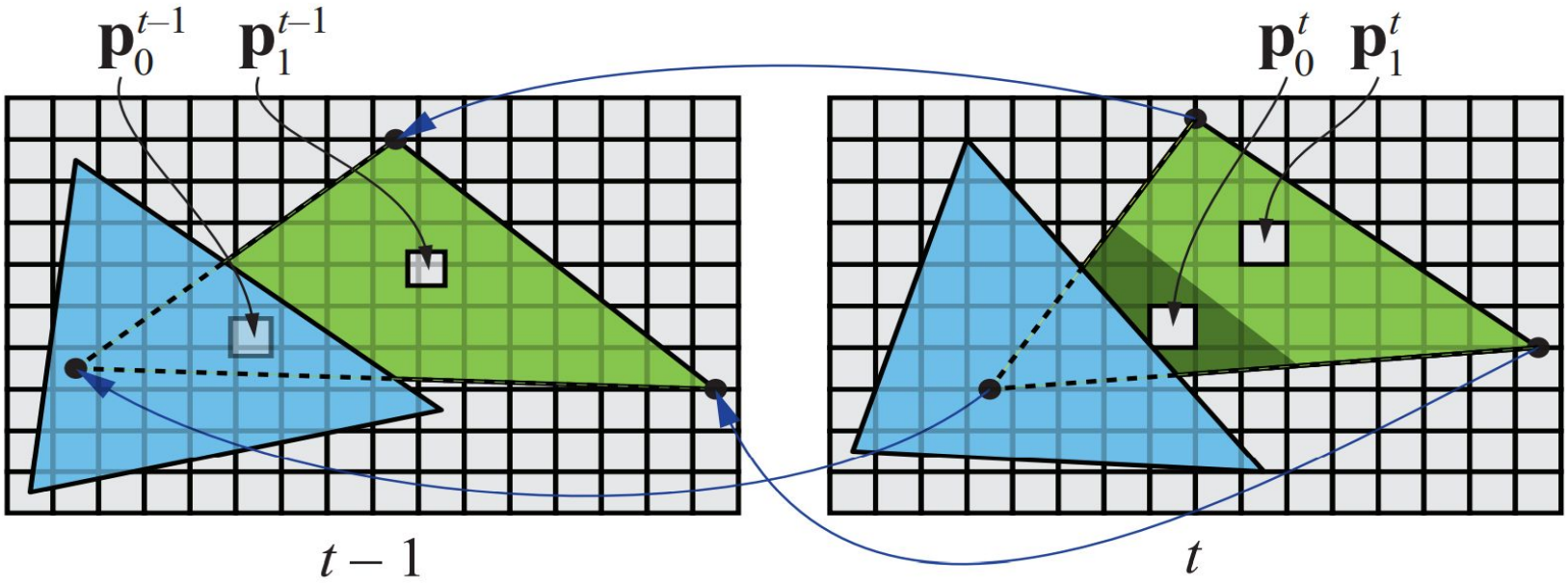
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

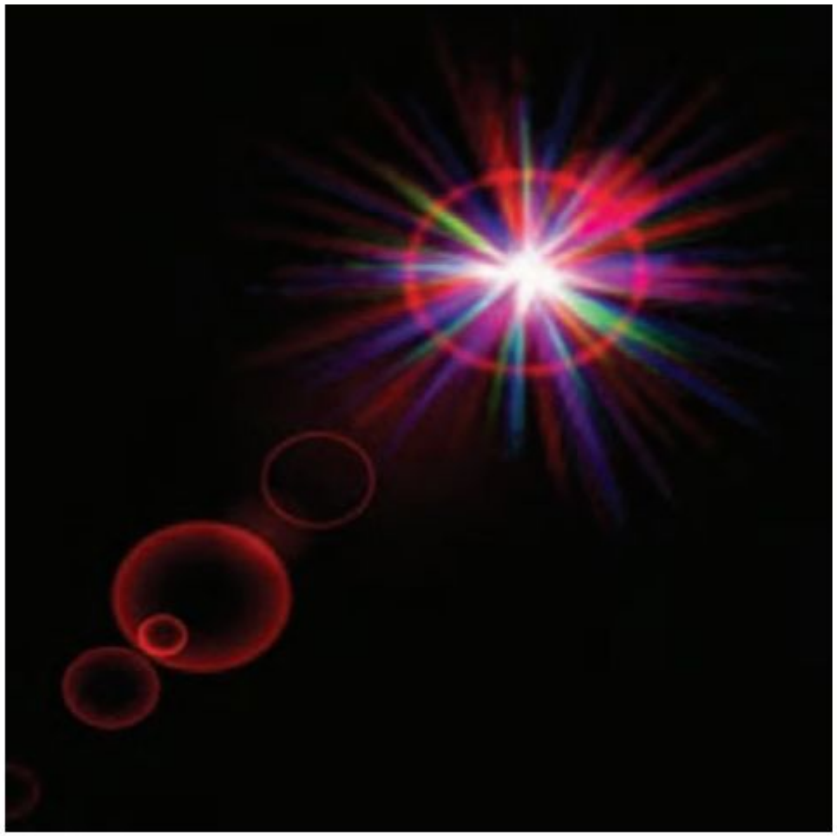
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

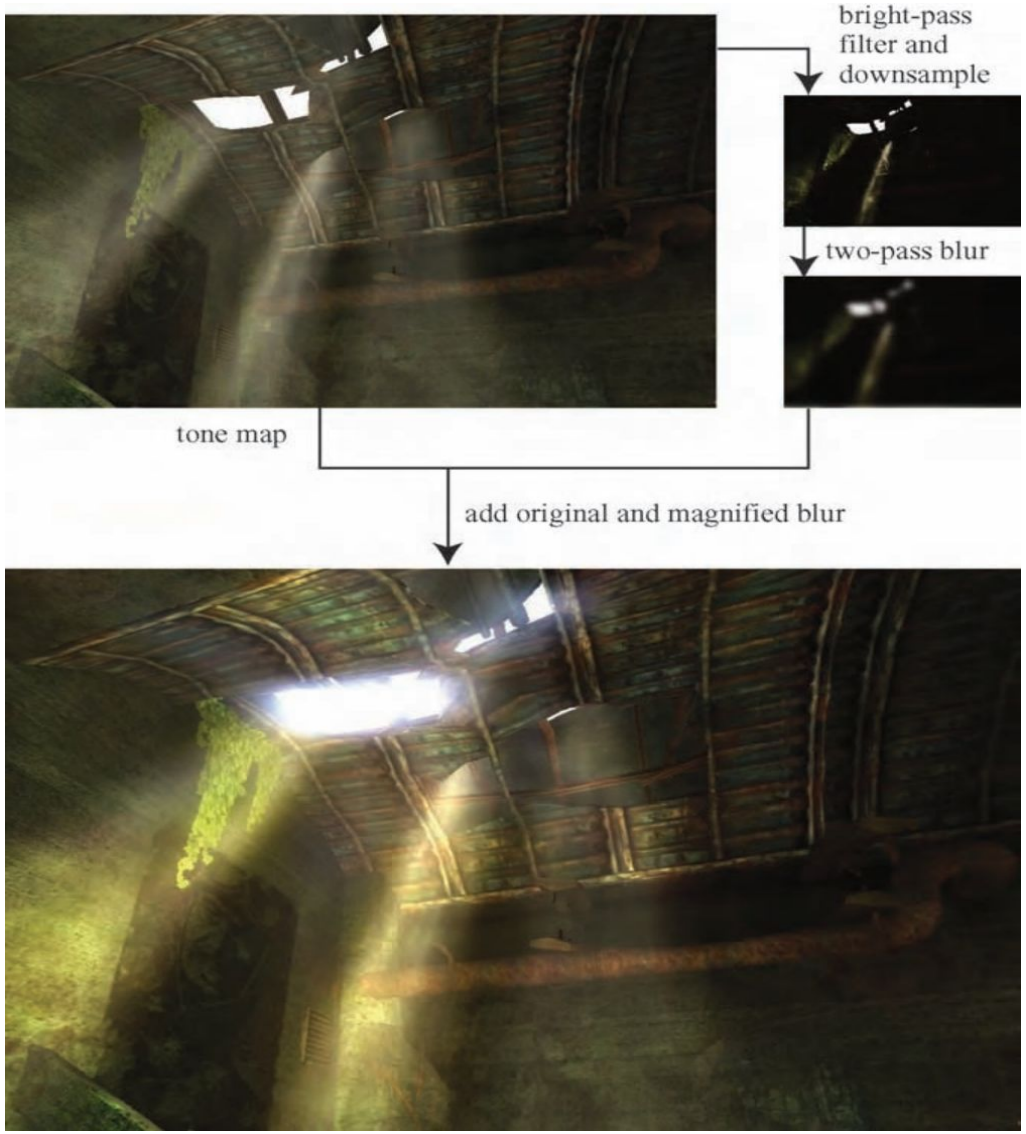
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

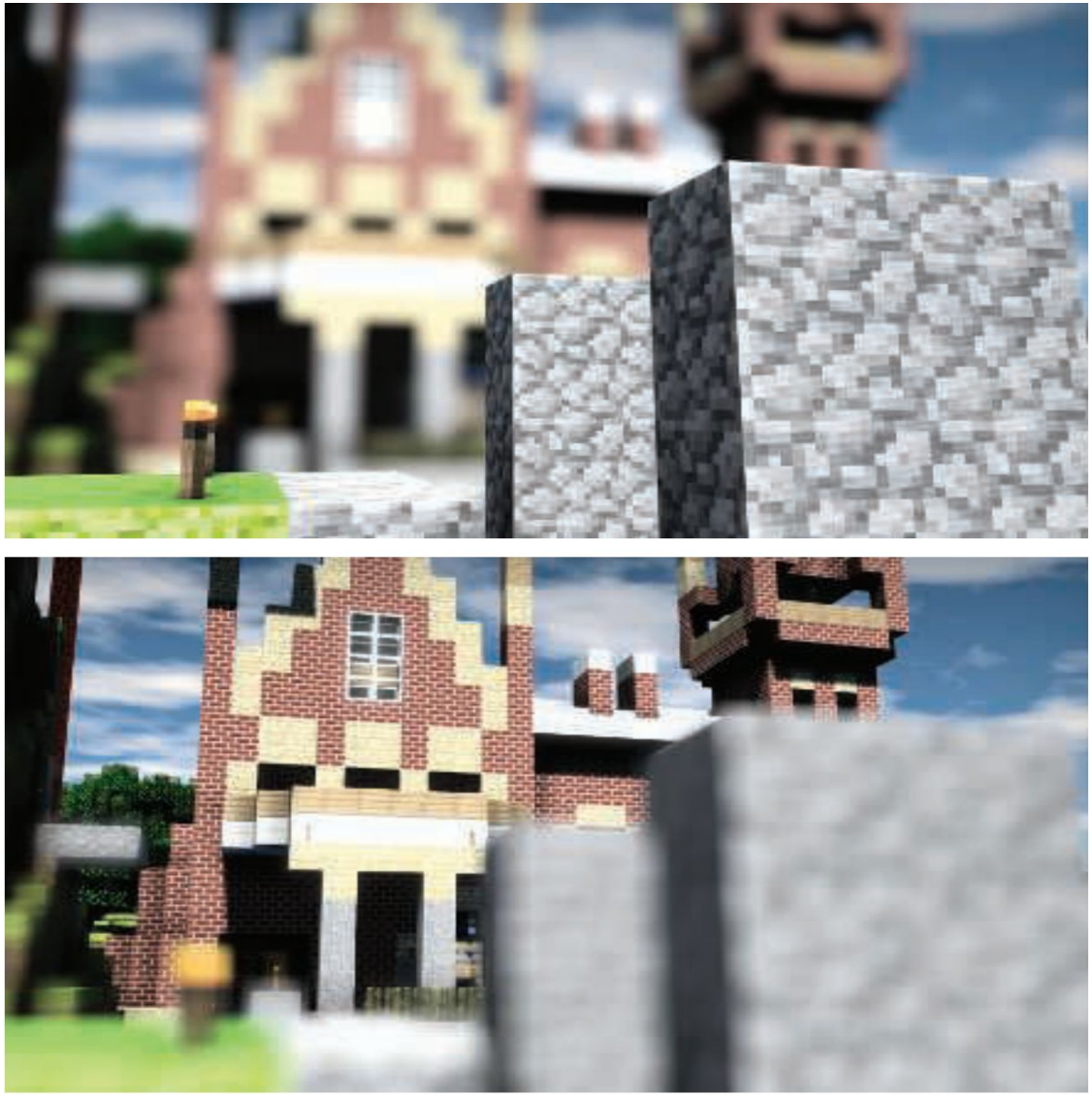
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

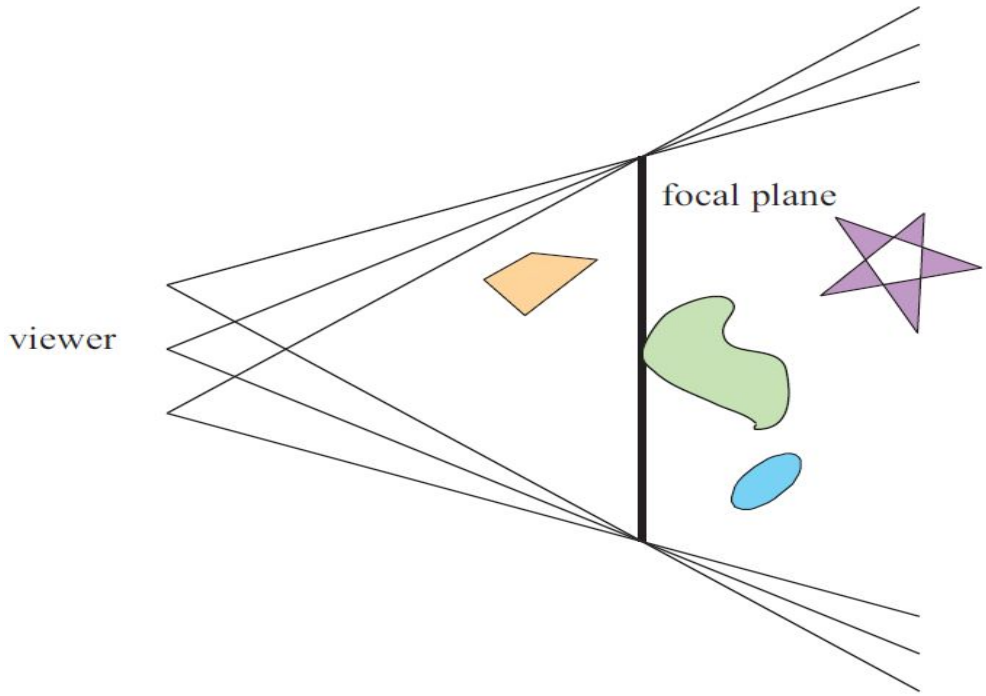


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

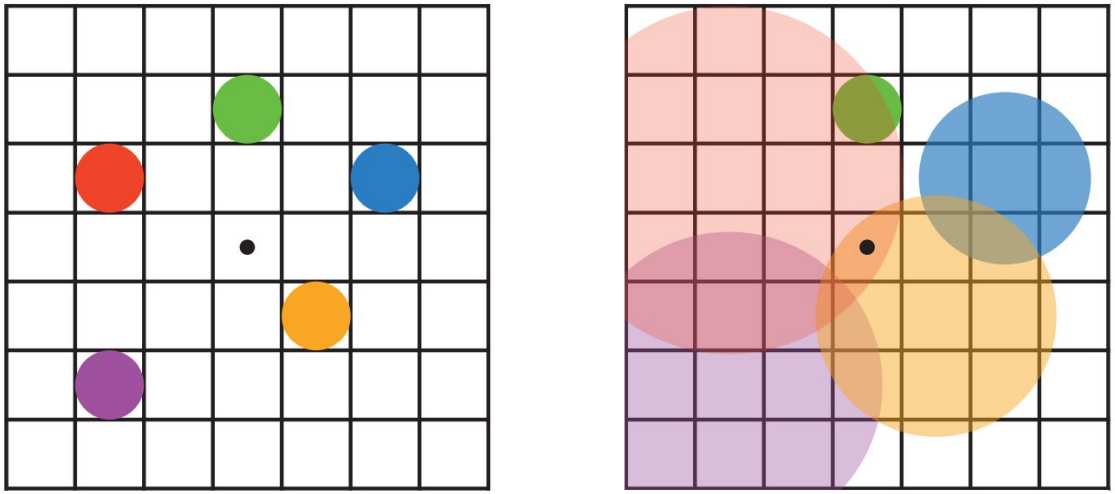
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

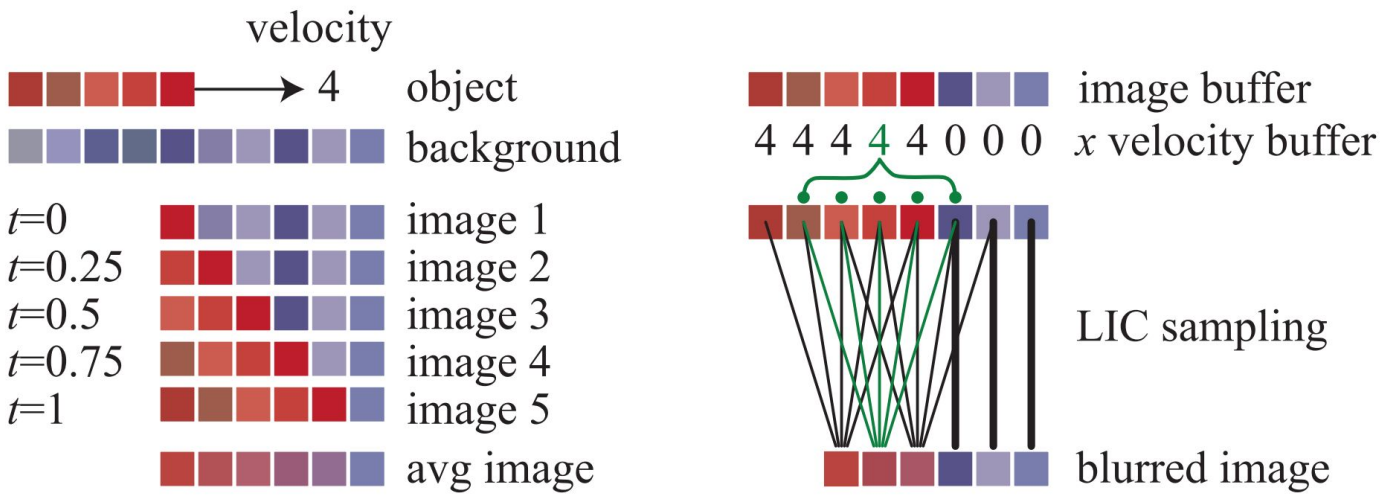
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

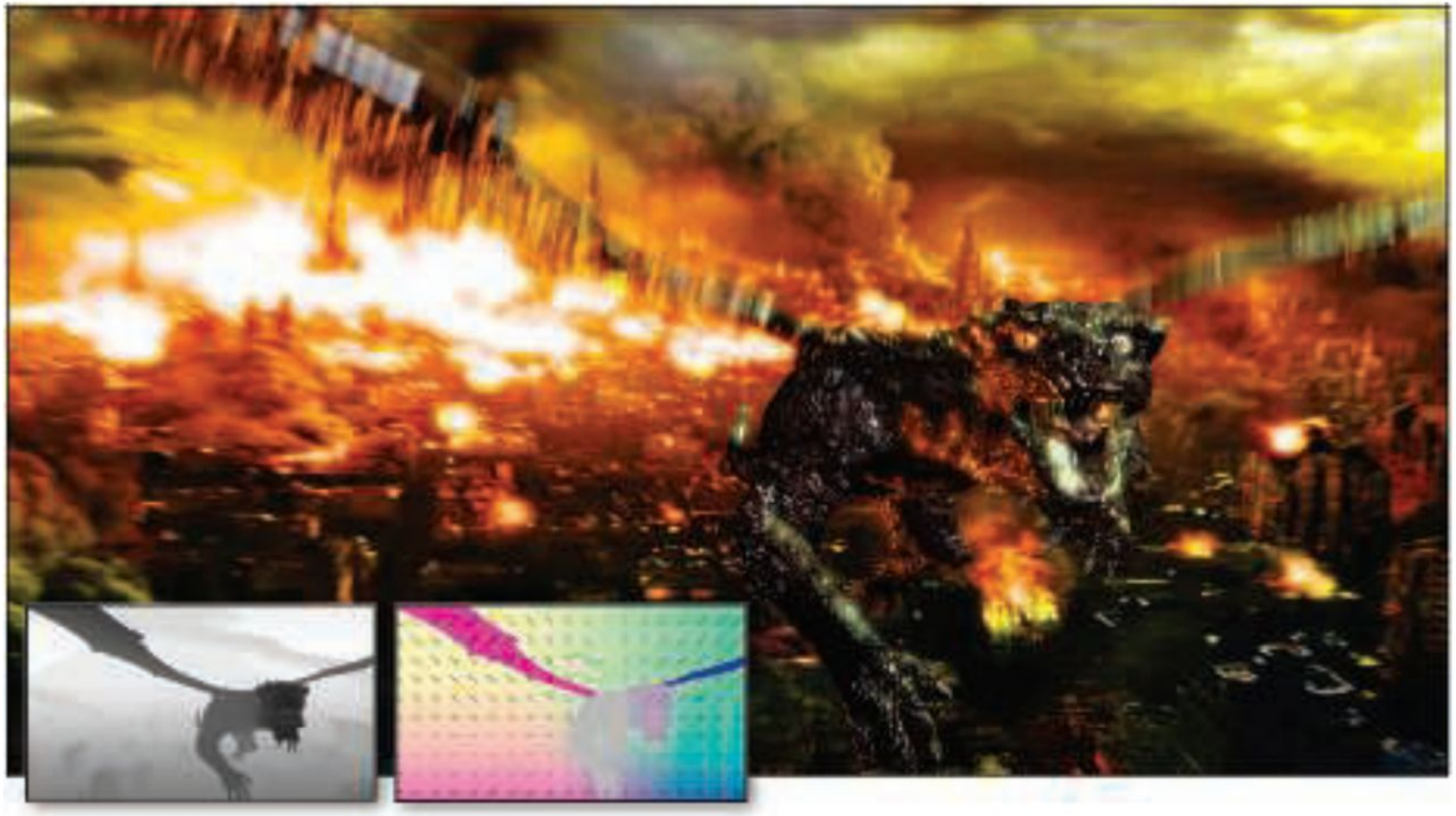
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

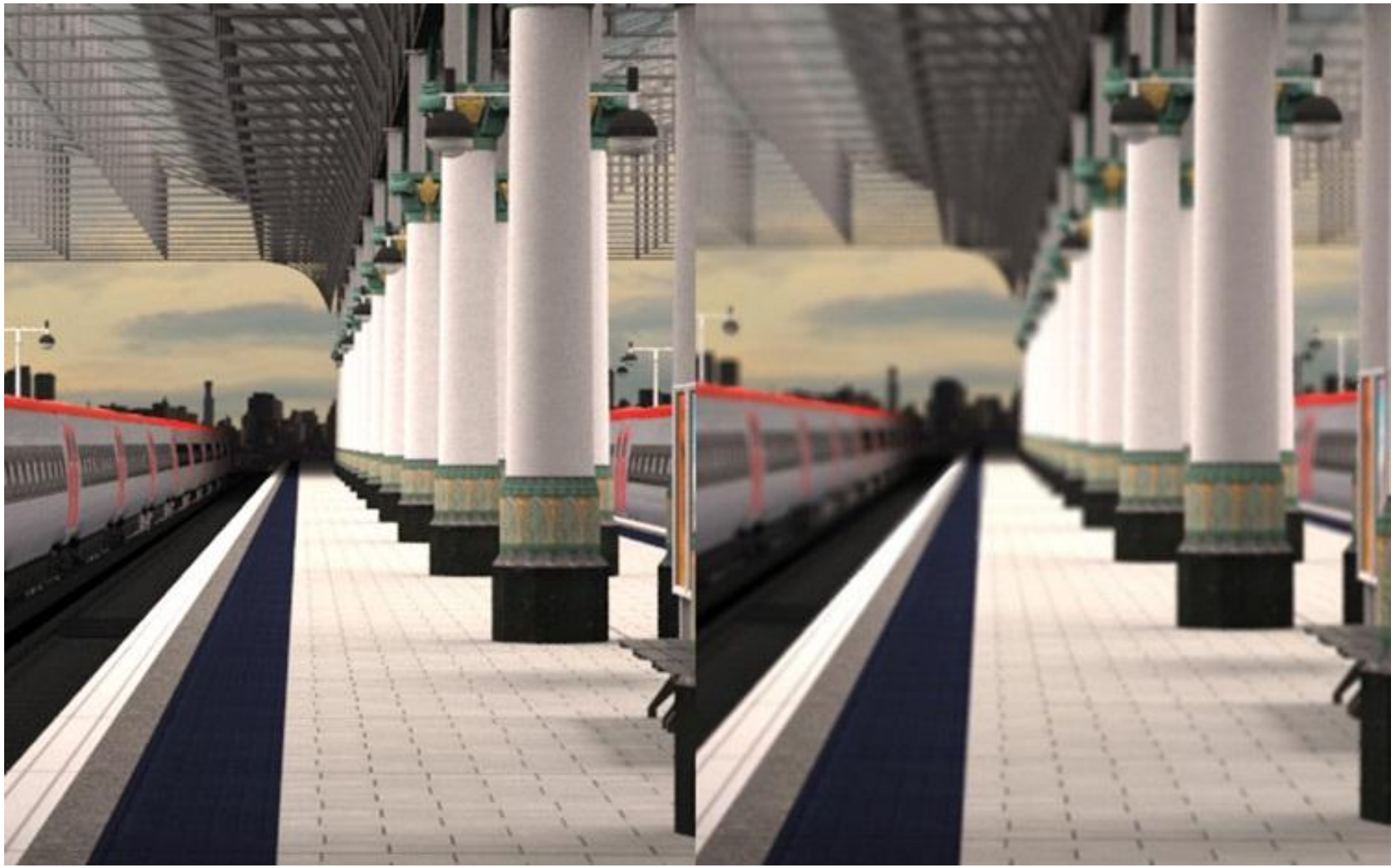
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

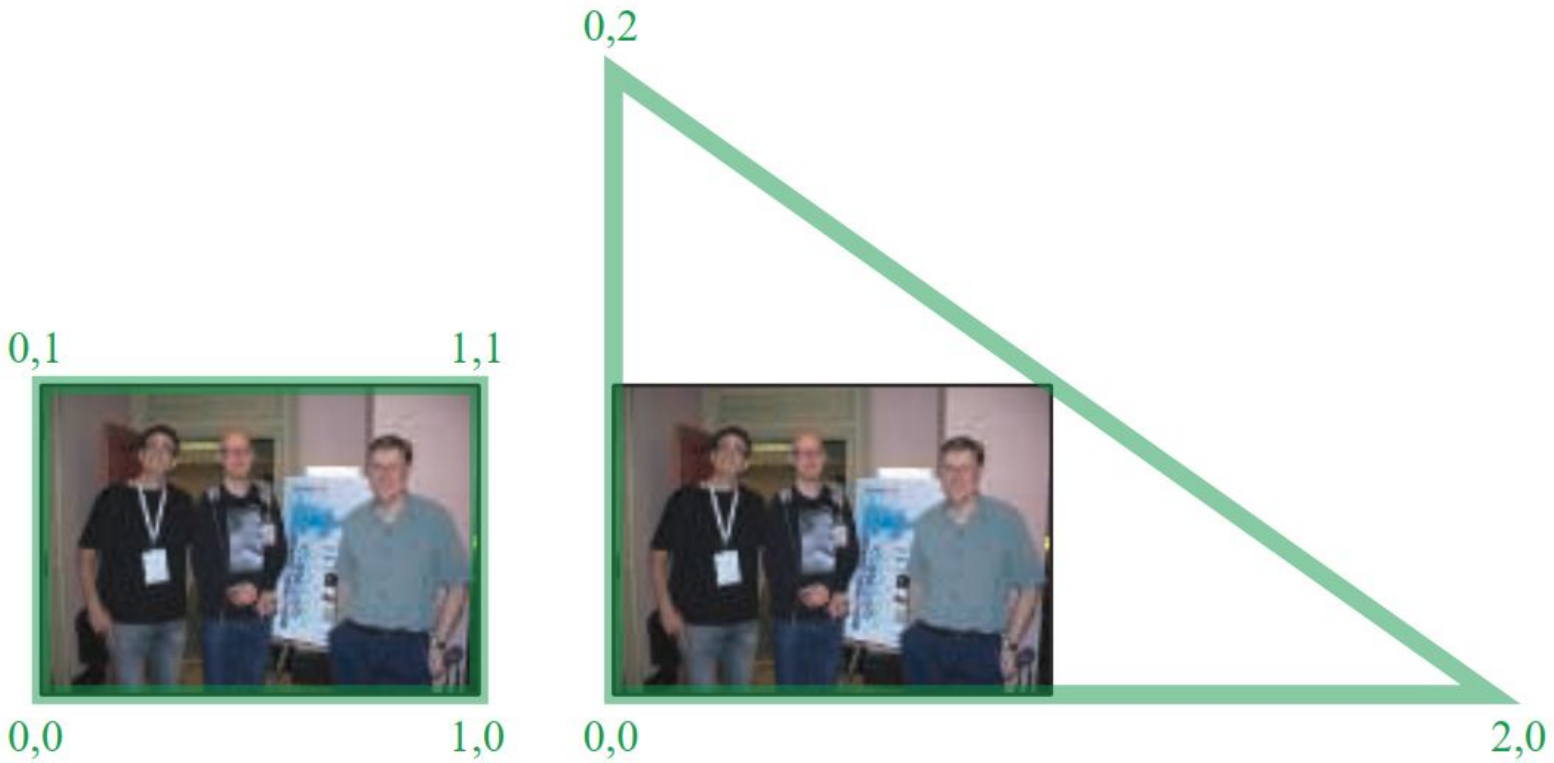
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.

Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

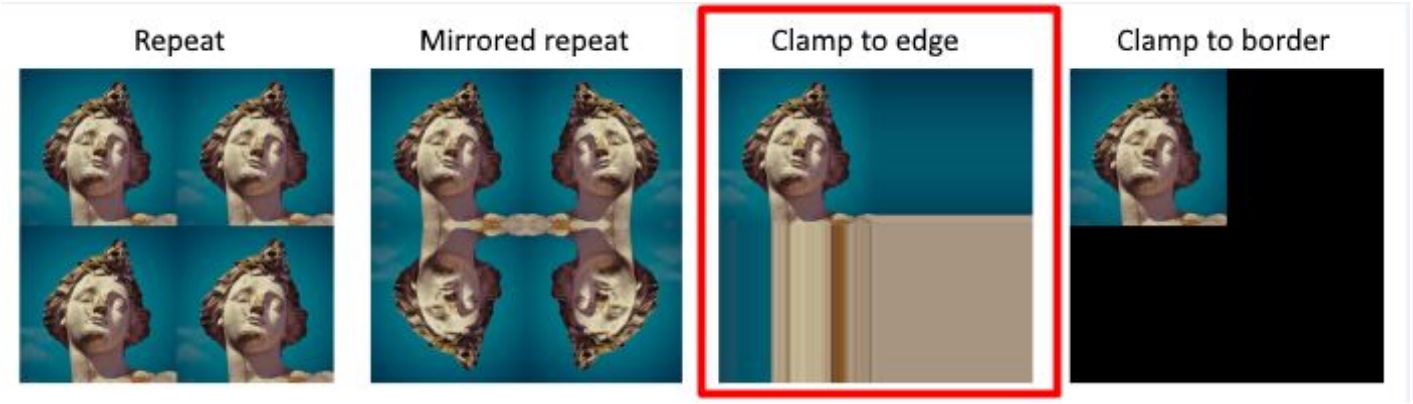
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

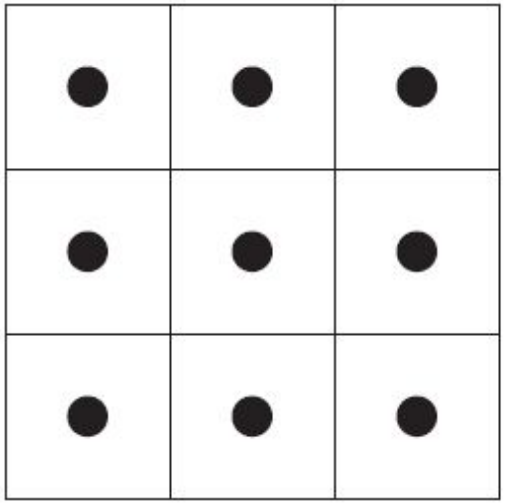
(c)

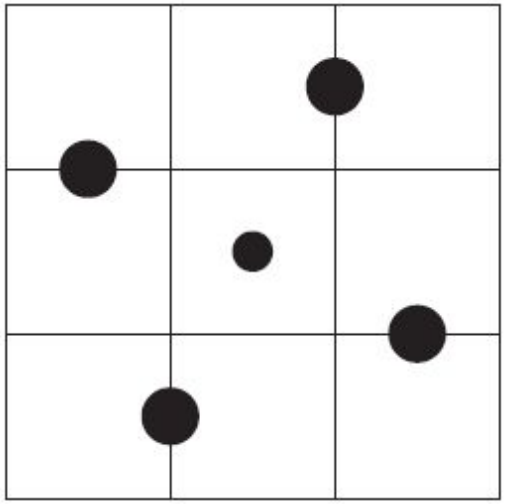
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

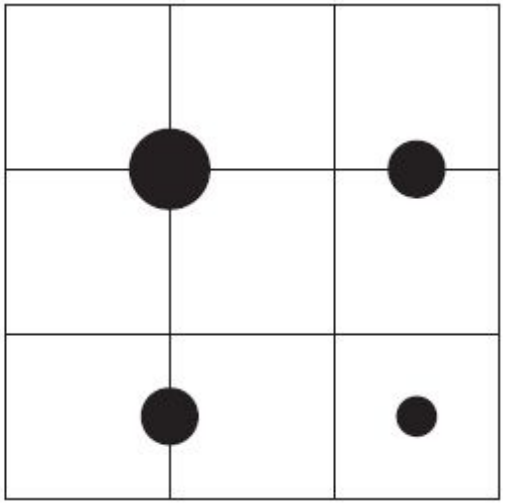
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

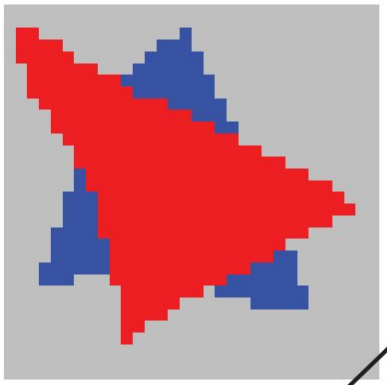
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

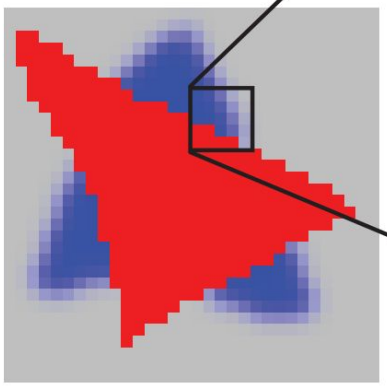
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

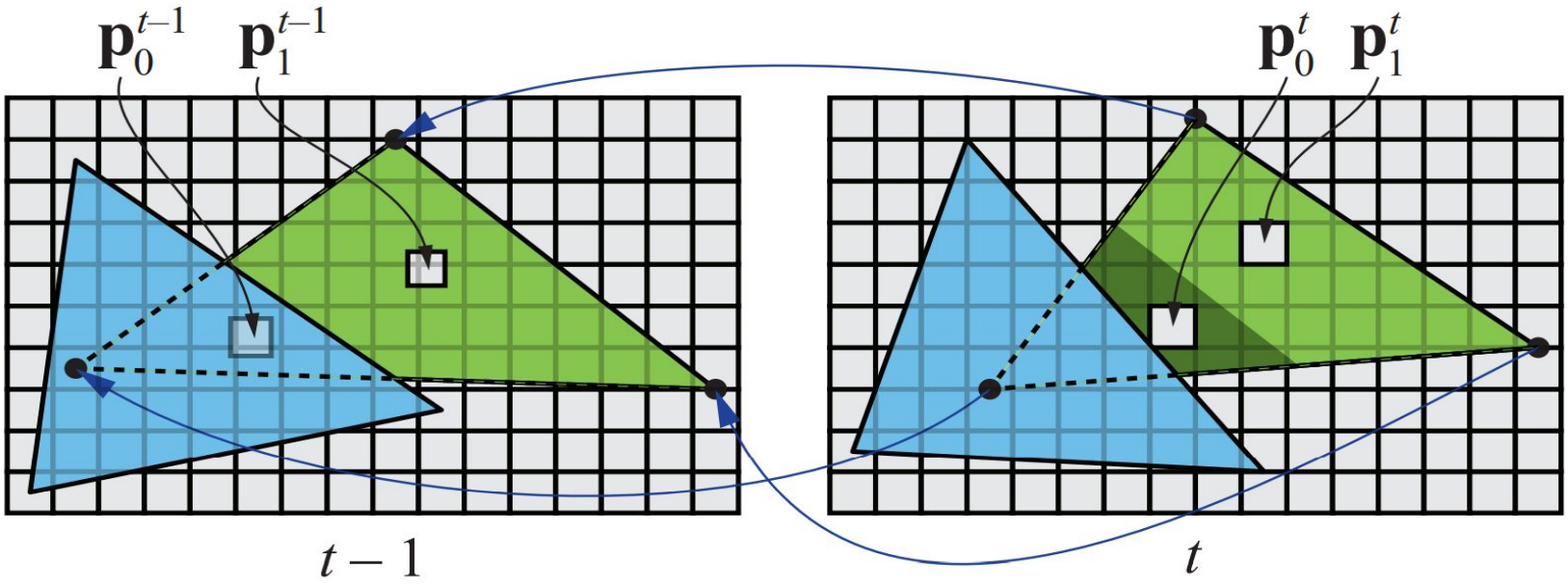
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

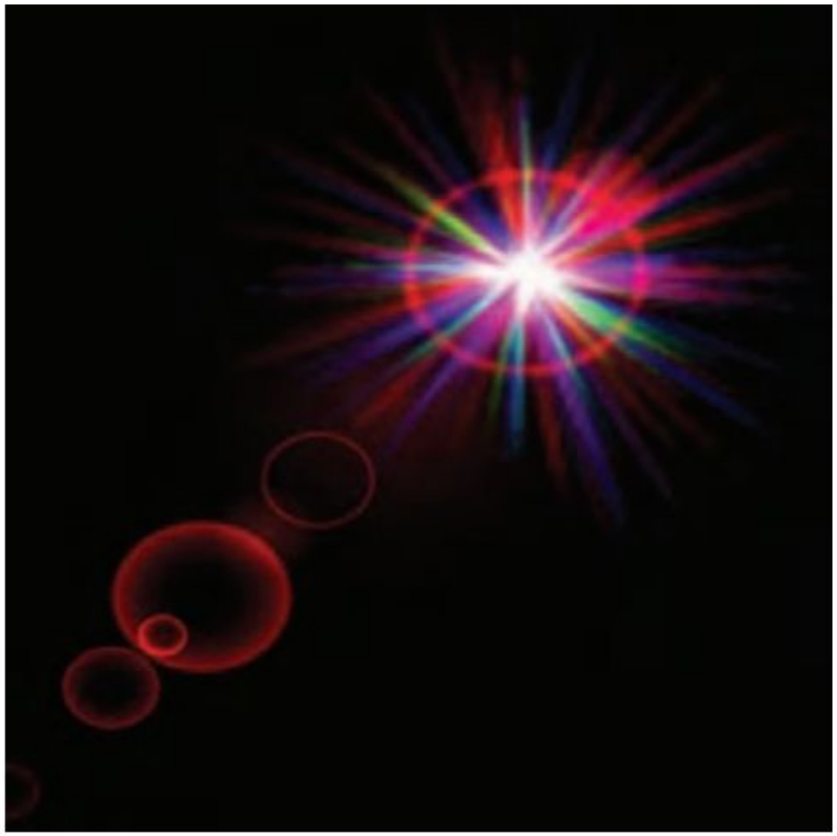
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

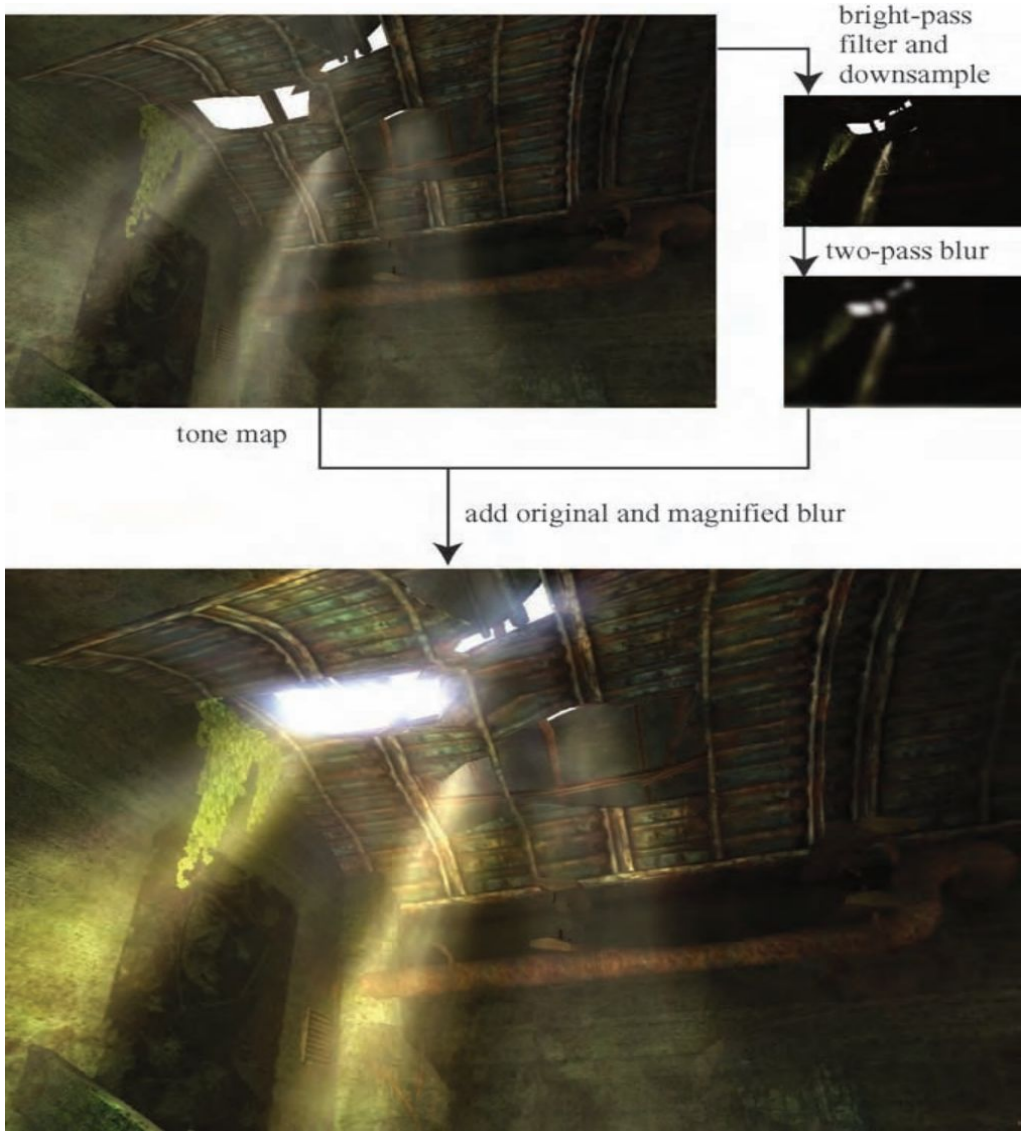
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

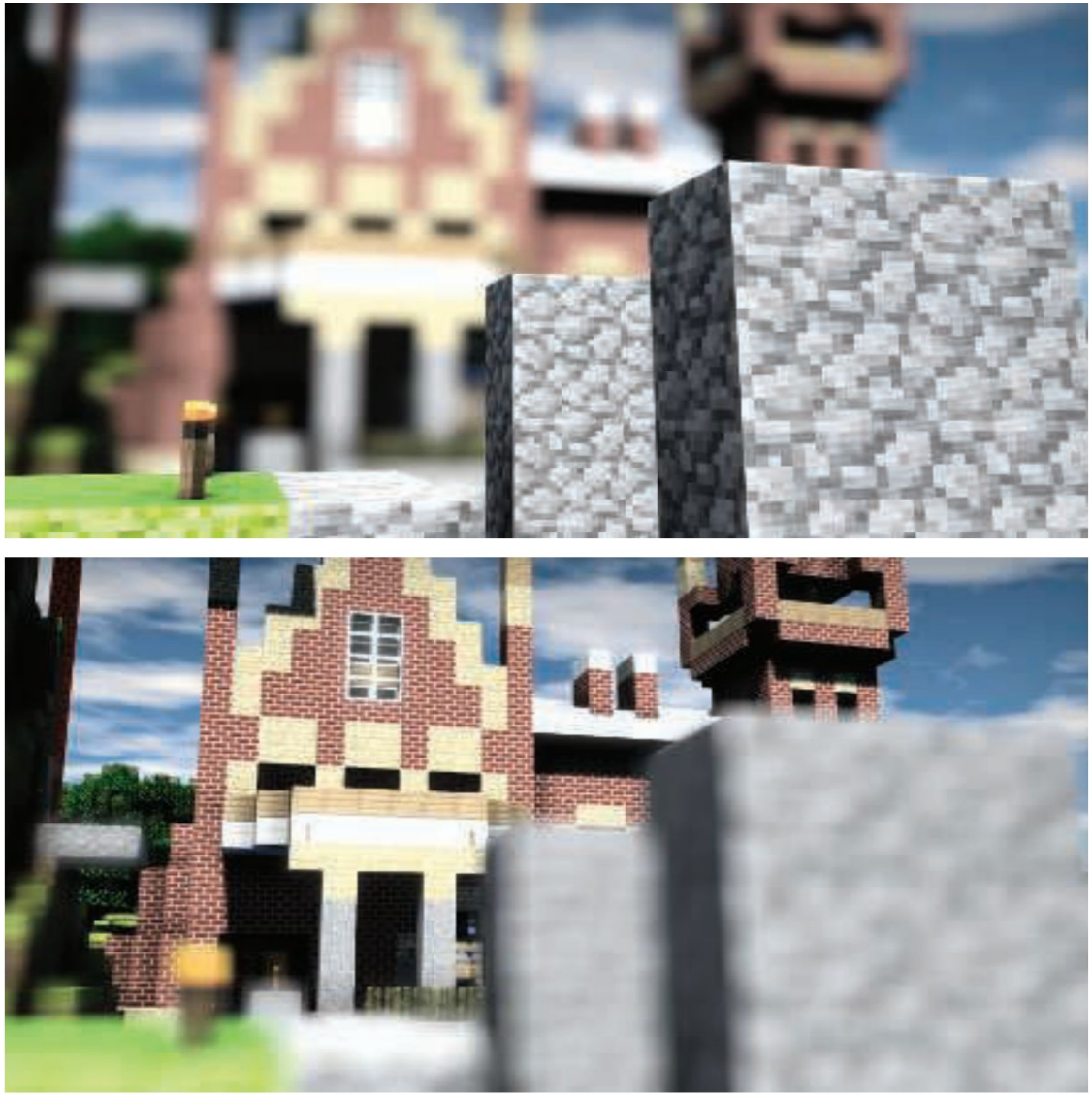
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

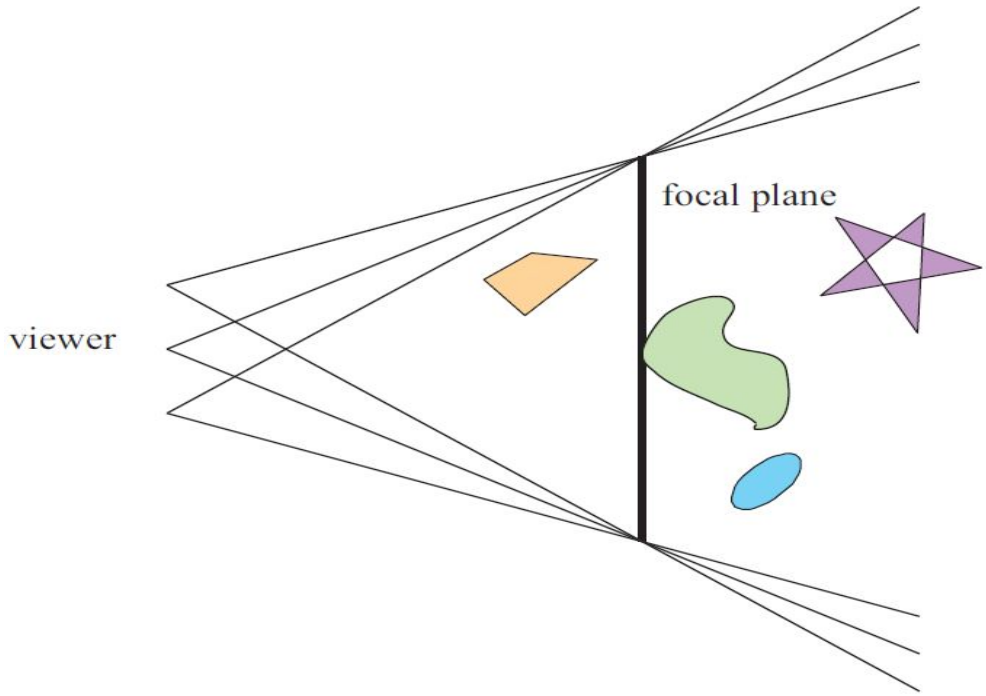


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

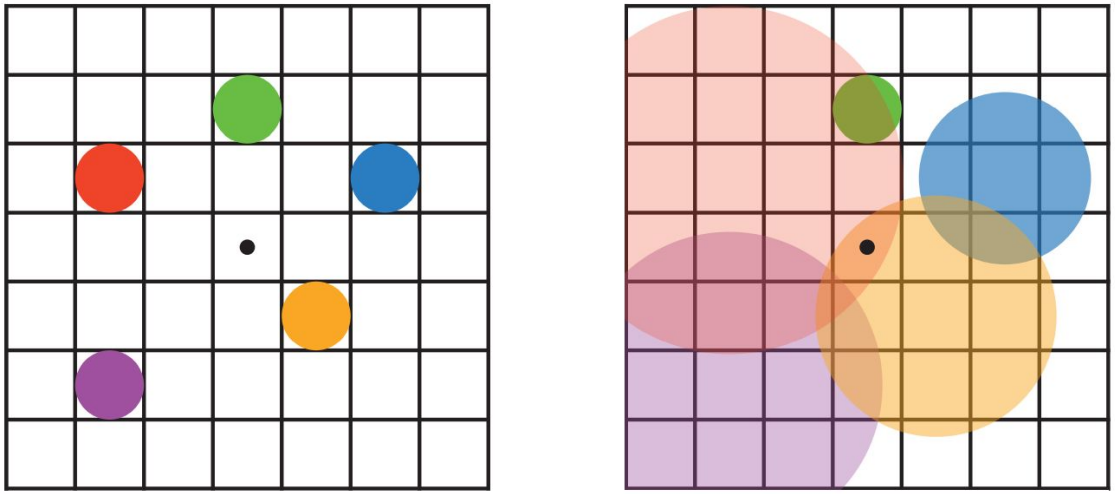
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

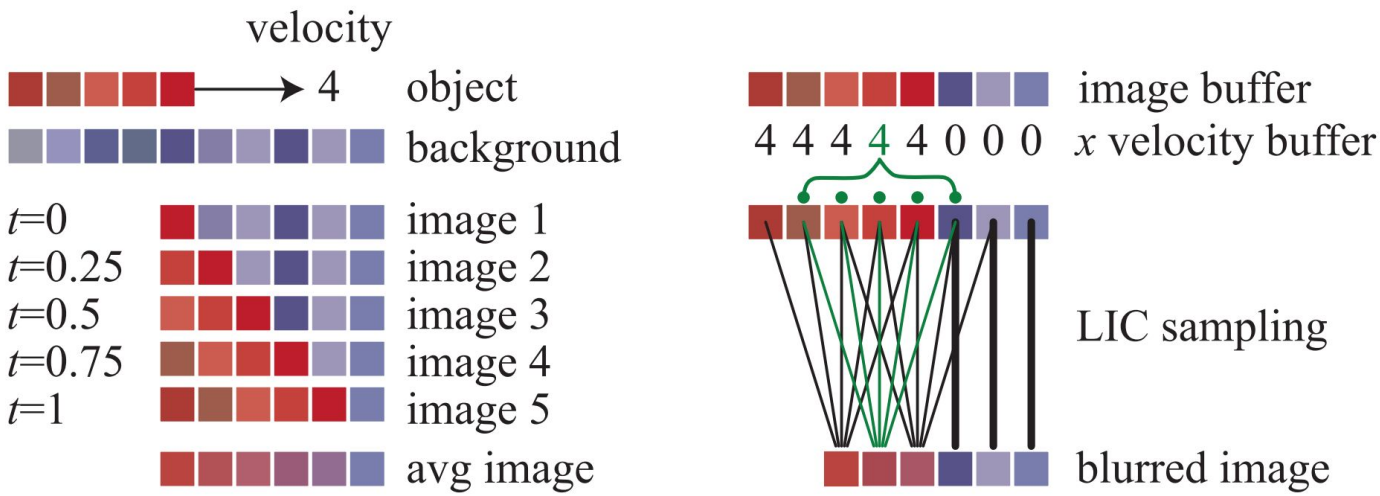
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

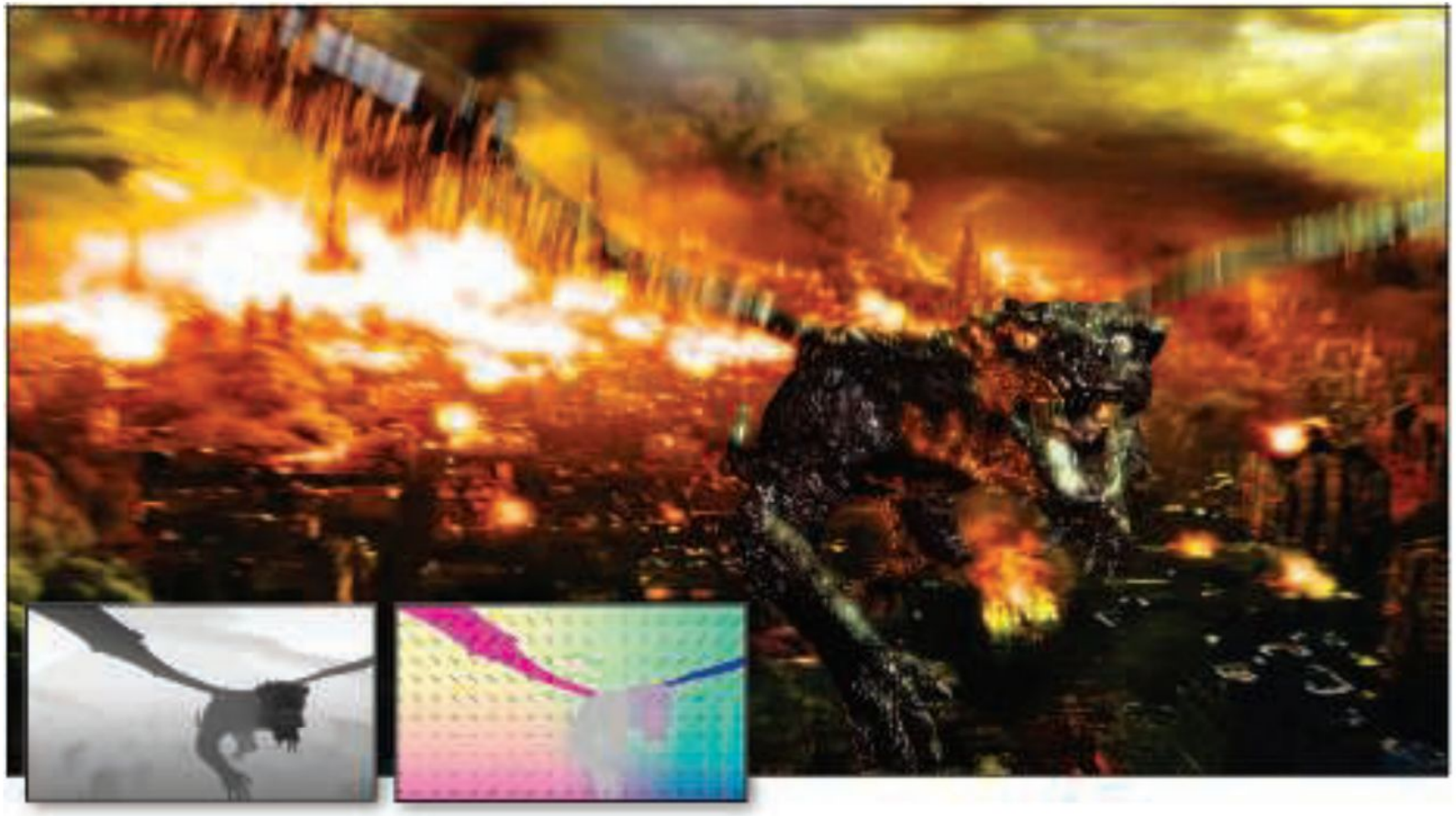
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

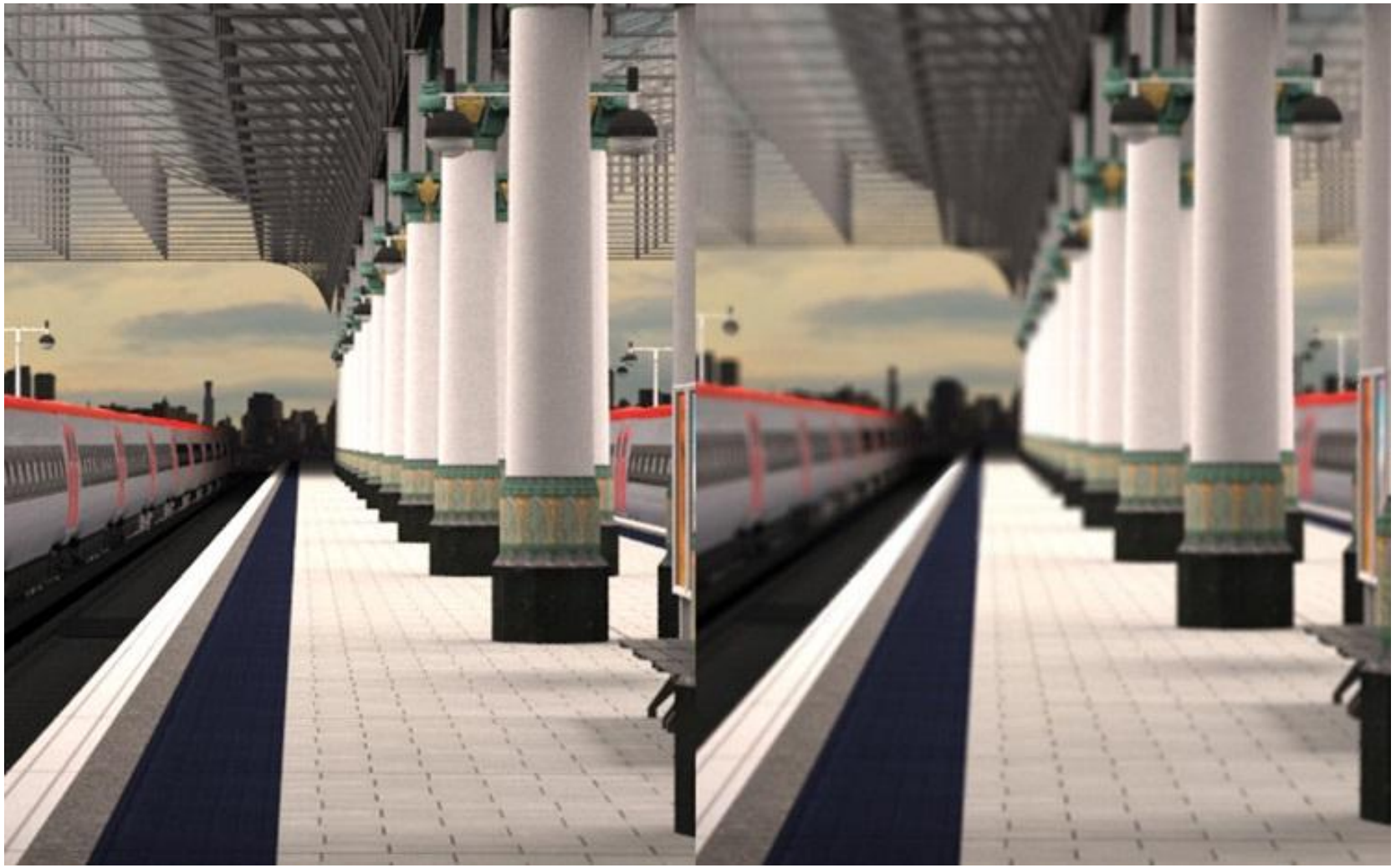
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

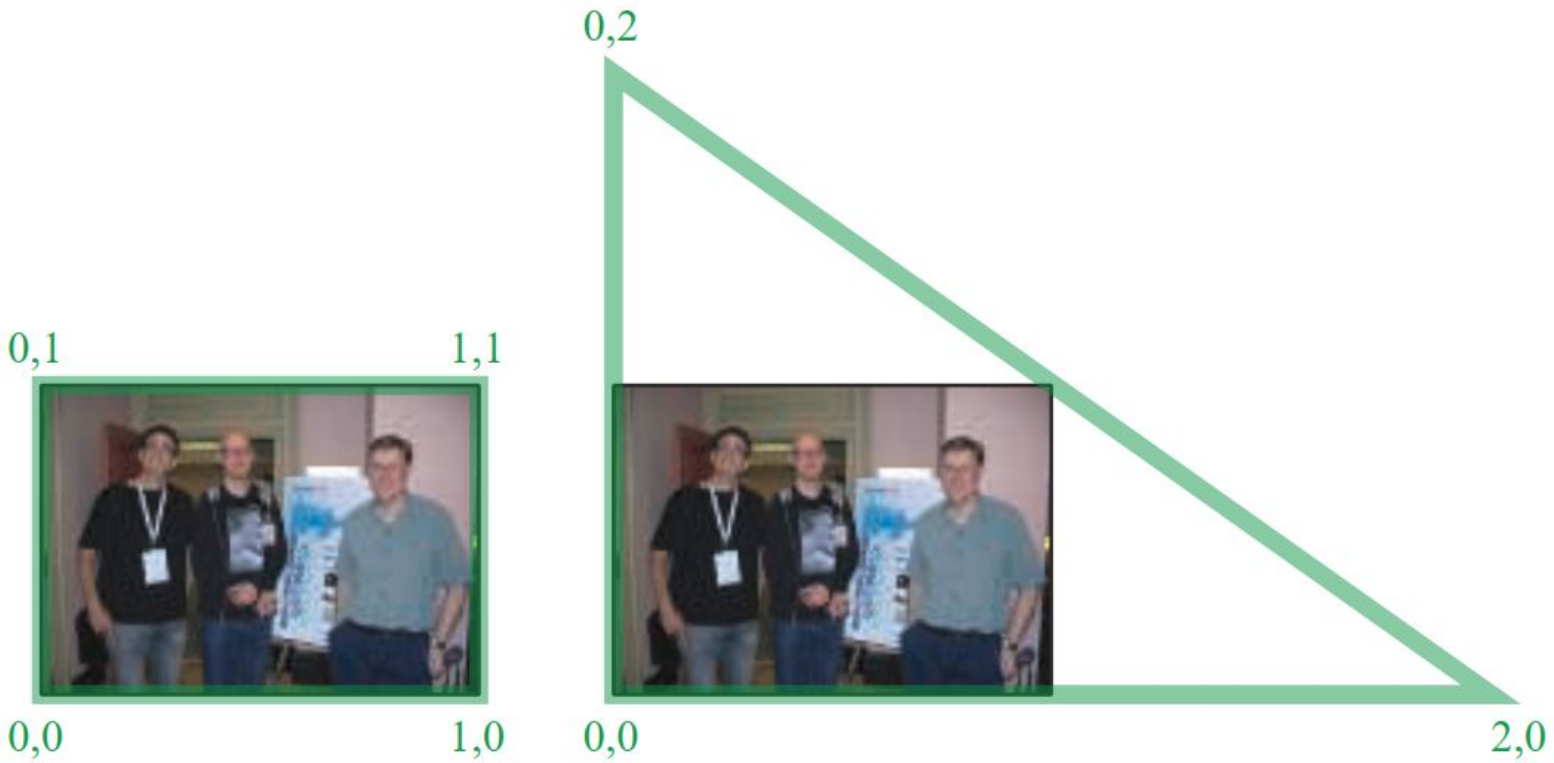
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.
Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

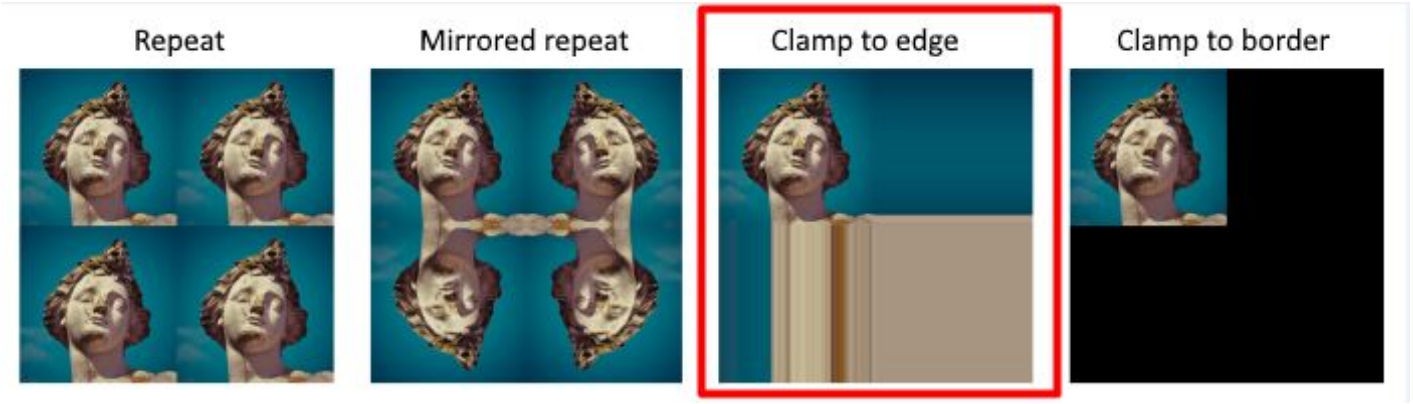
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

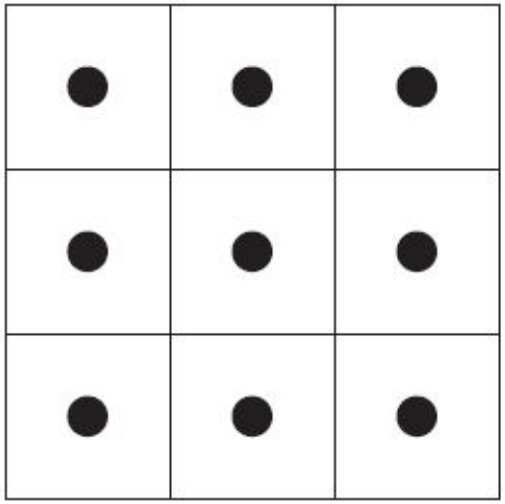
(c)

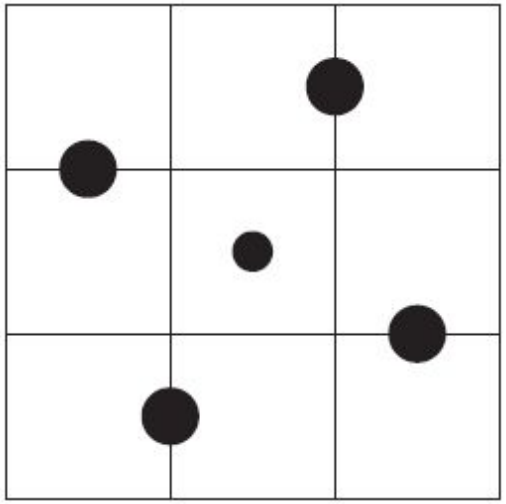
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

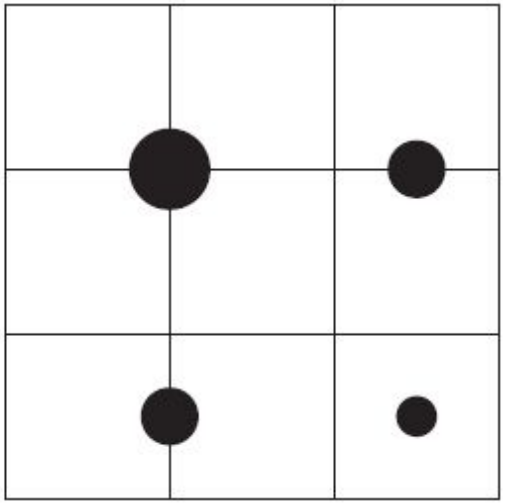
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

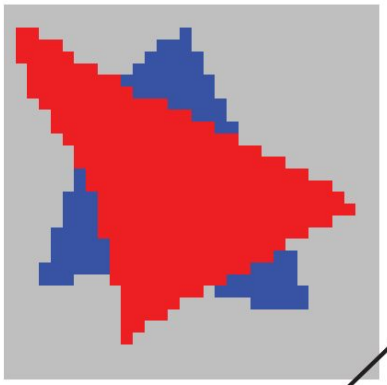
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

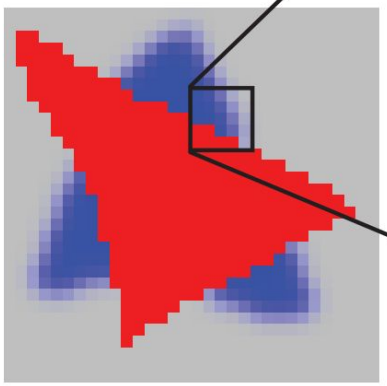
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

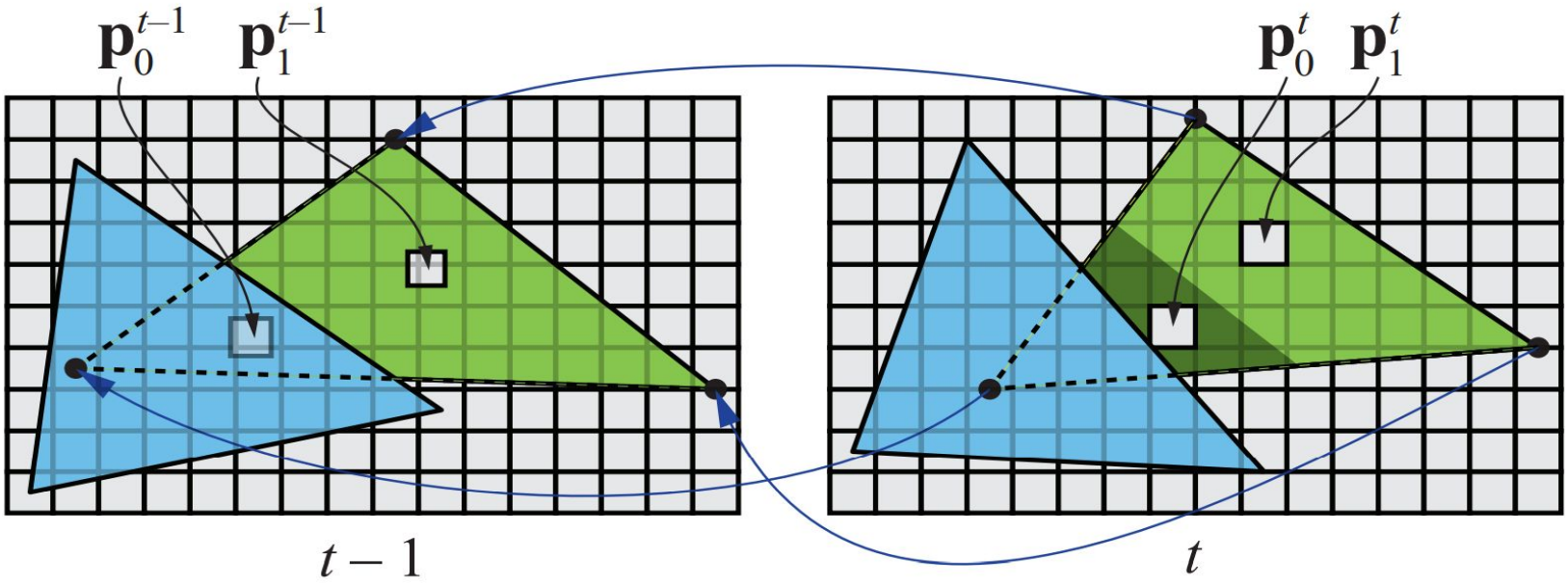
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

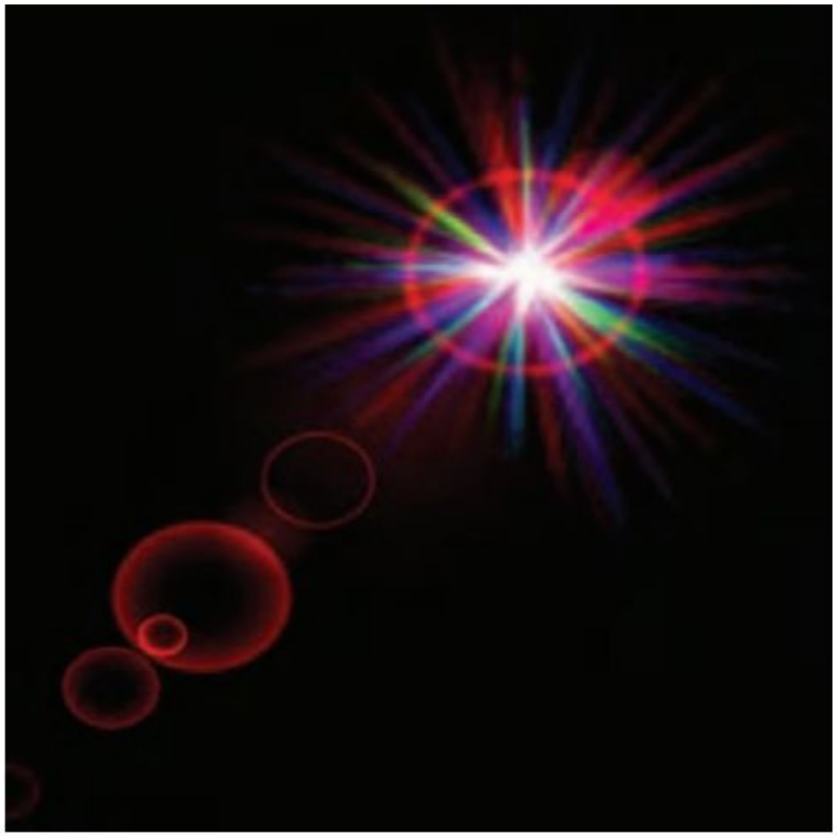
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

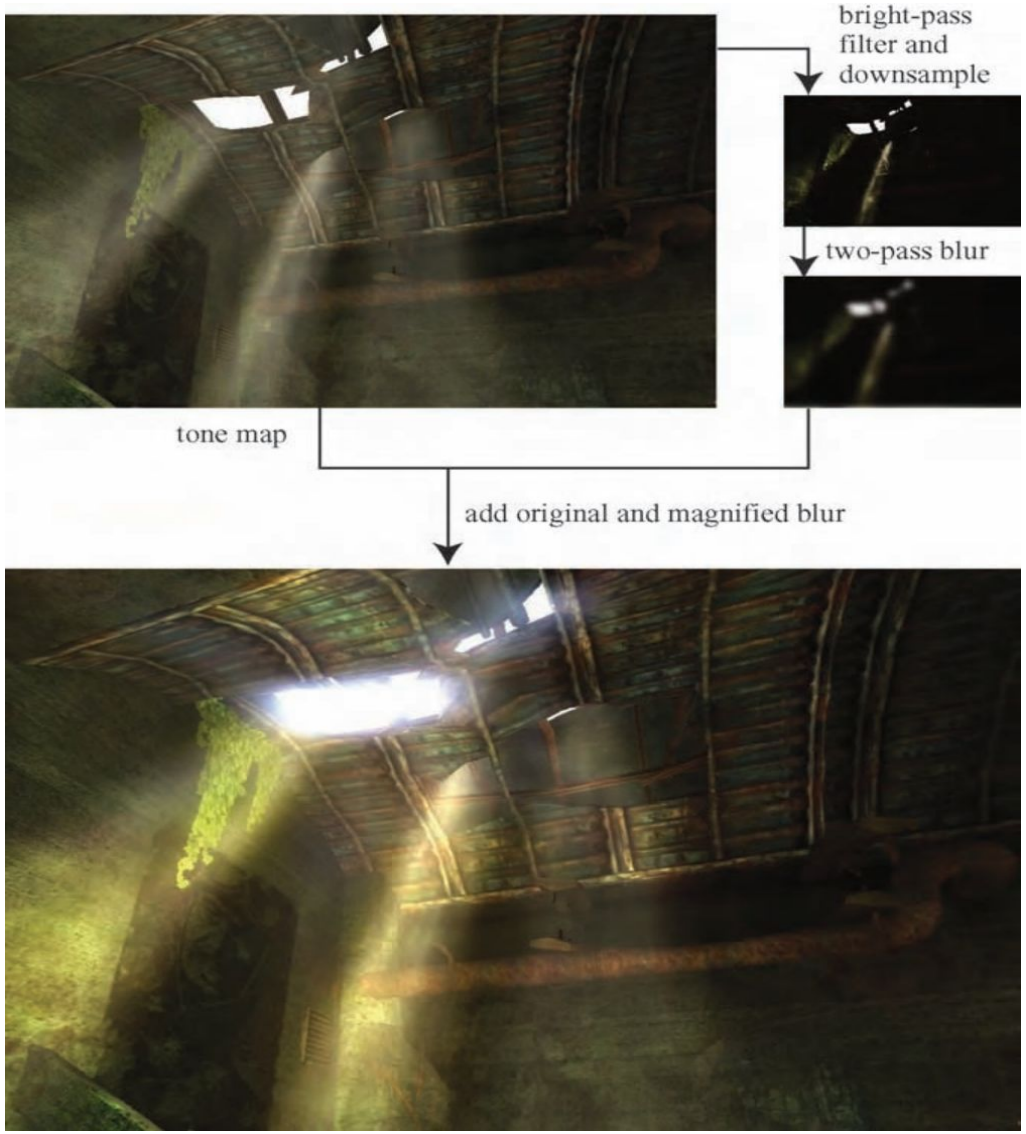
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

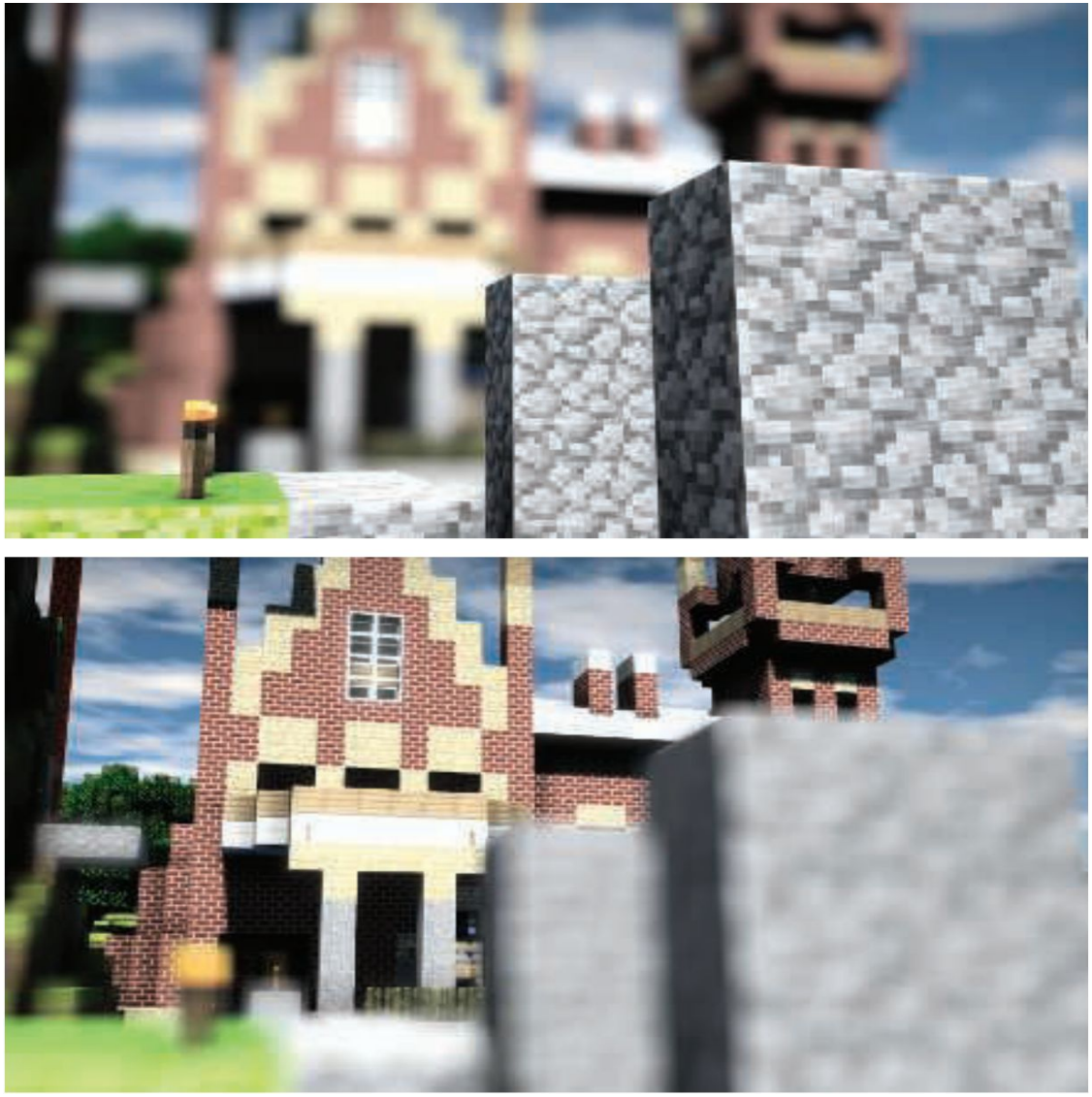
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

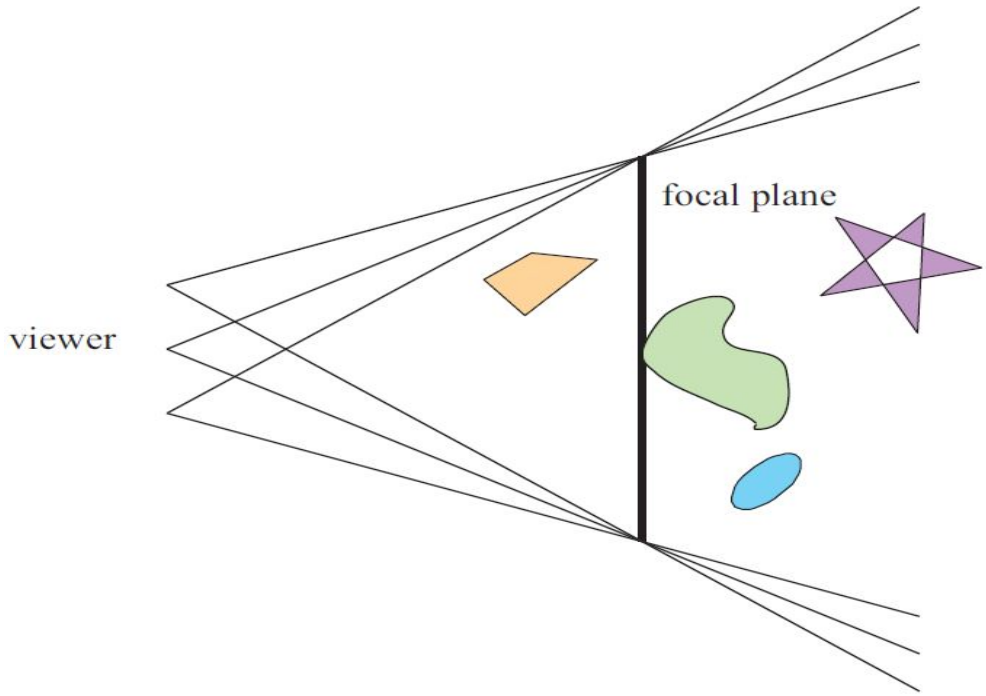


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

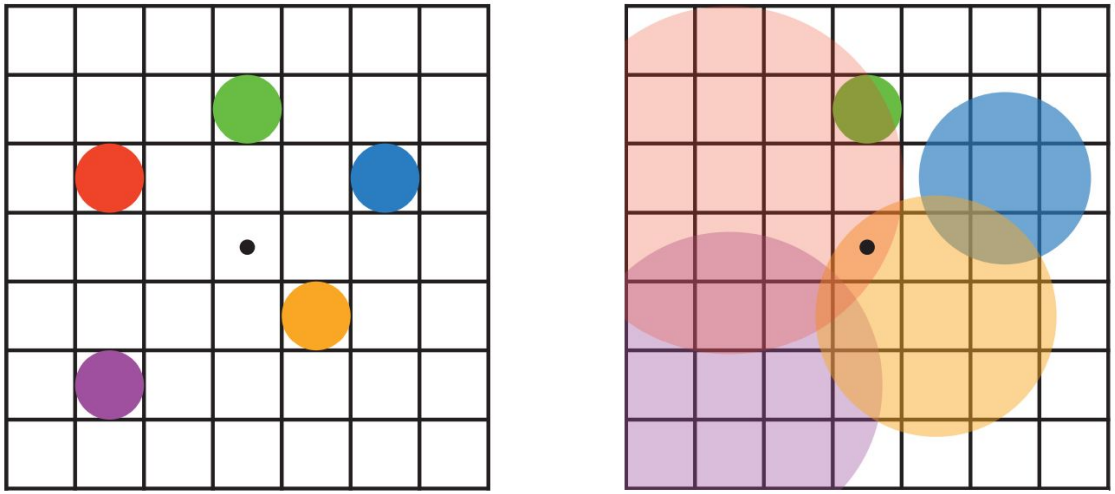
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

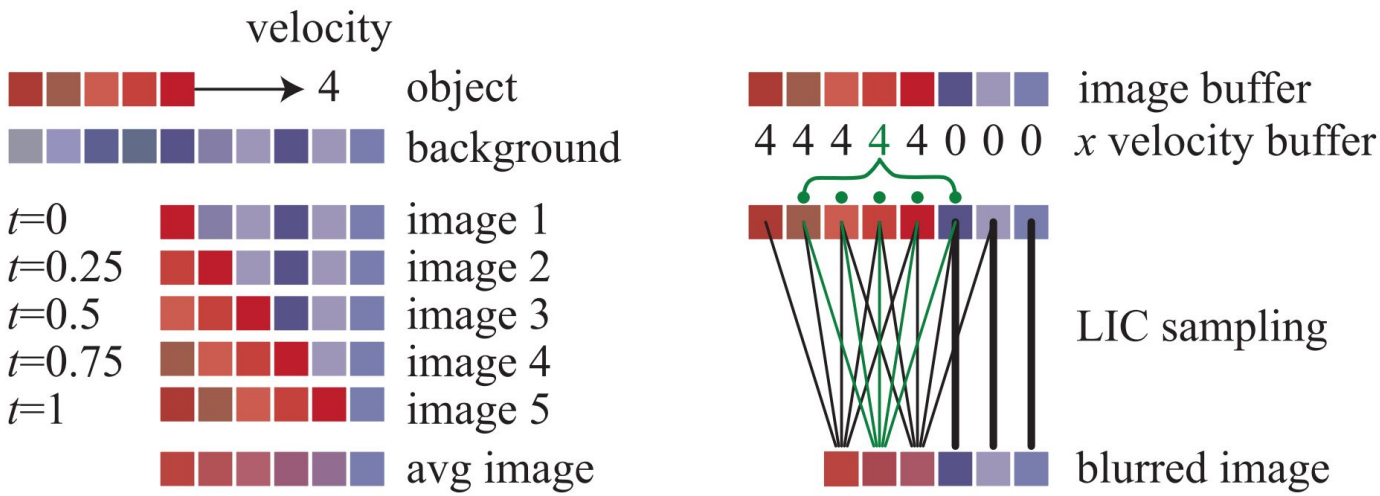
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

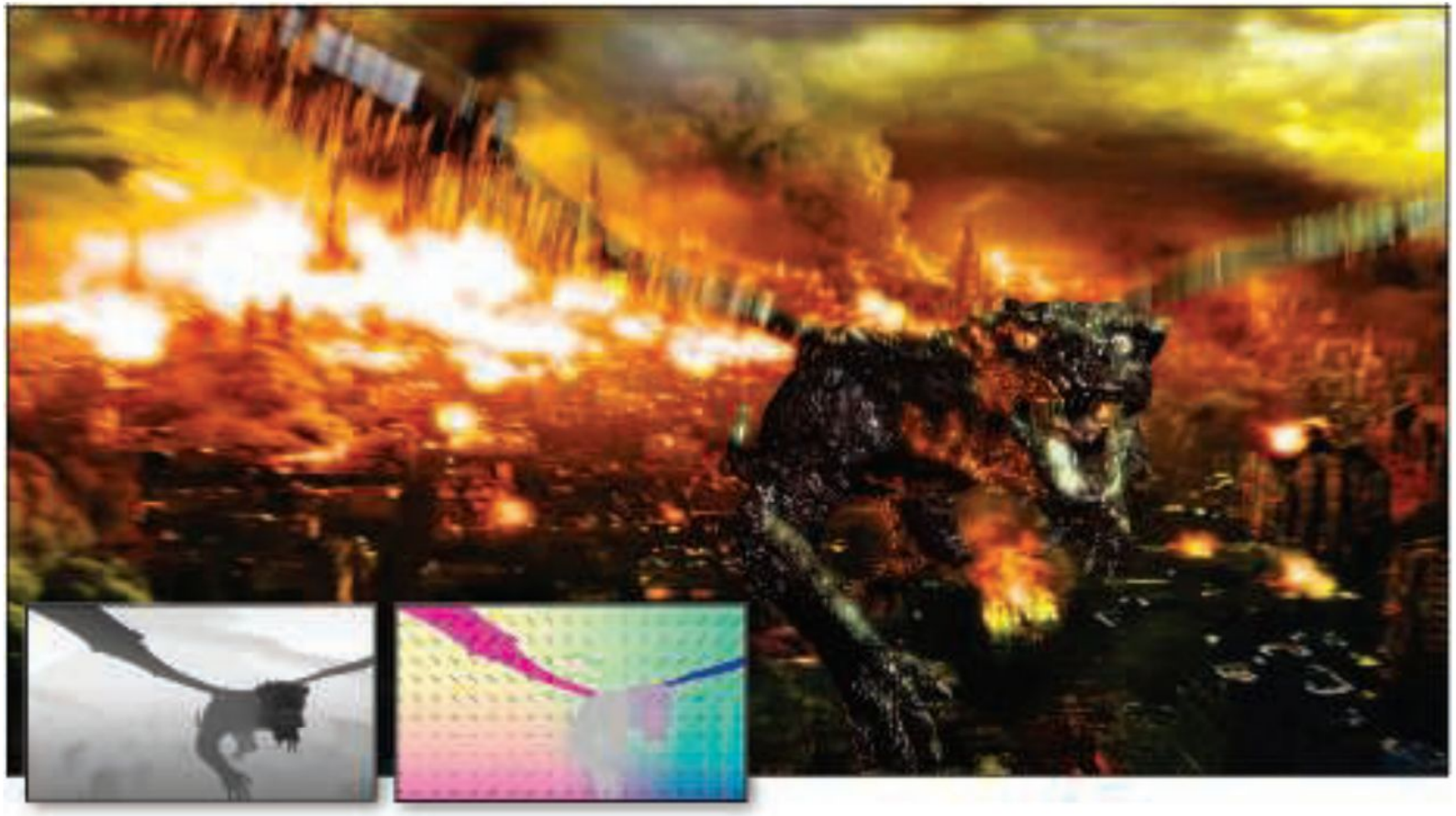
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

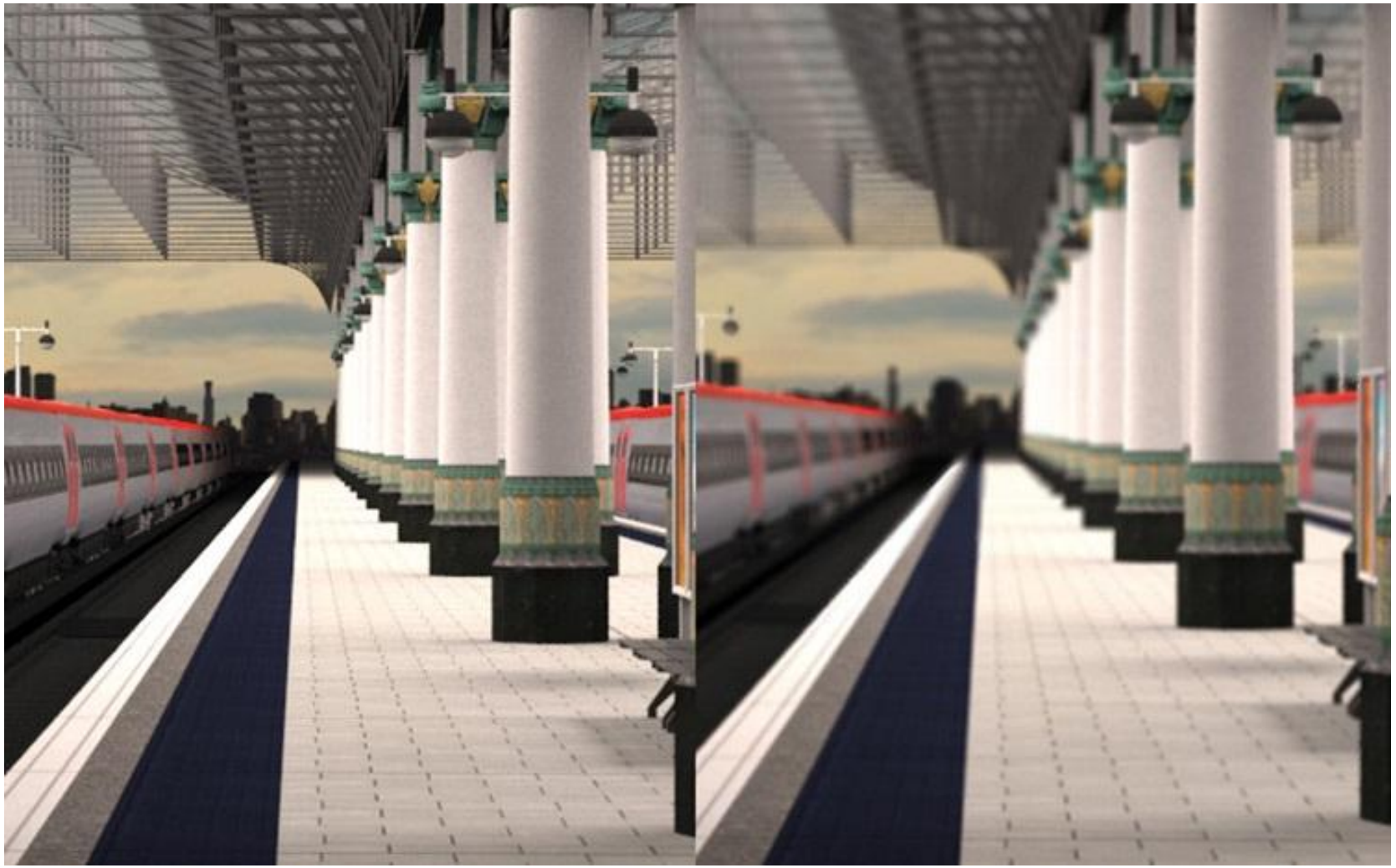
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

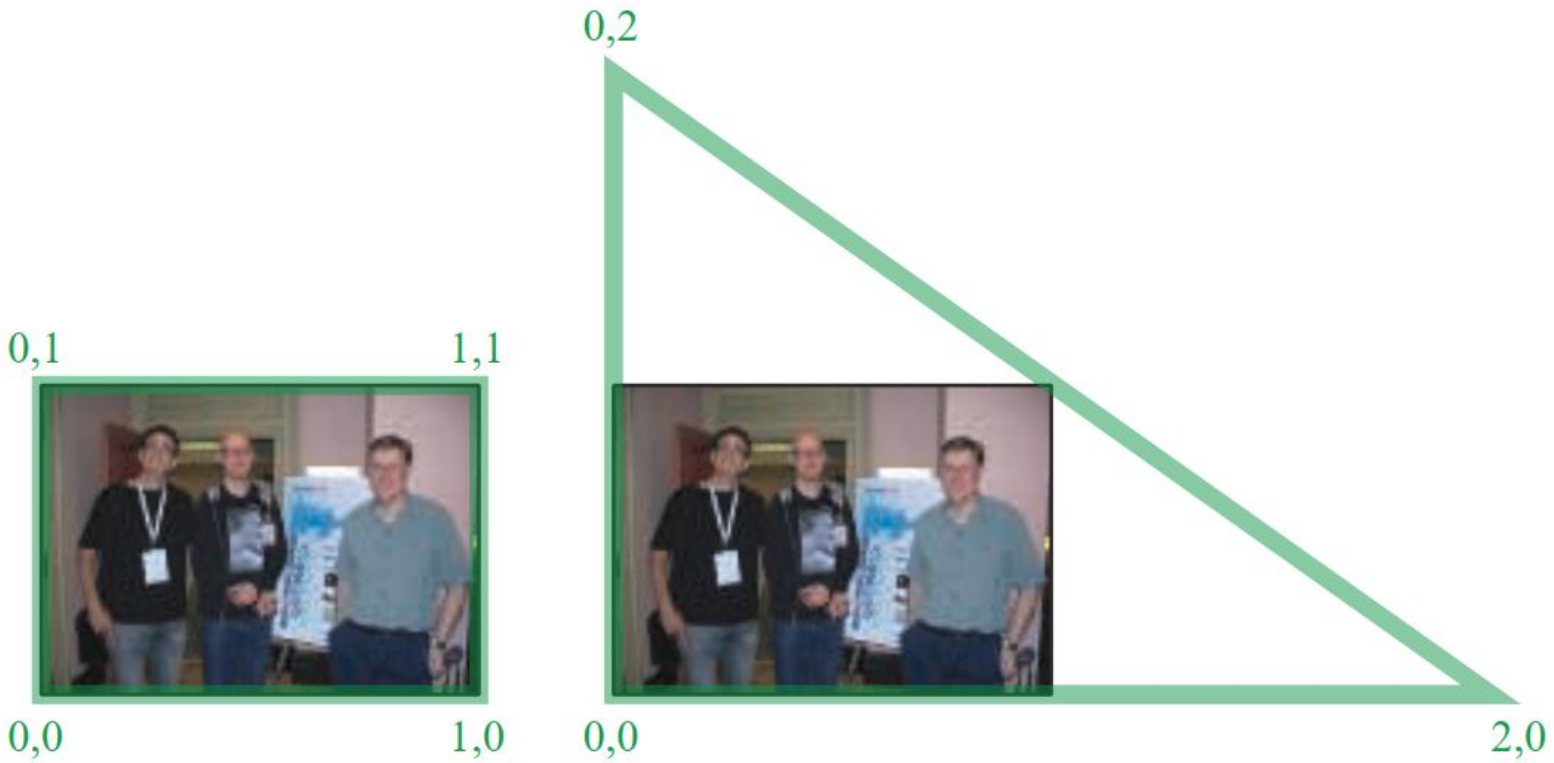
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.
Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

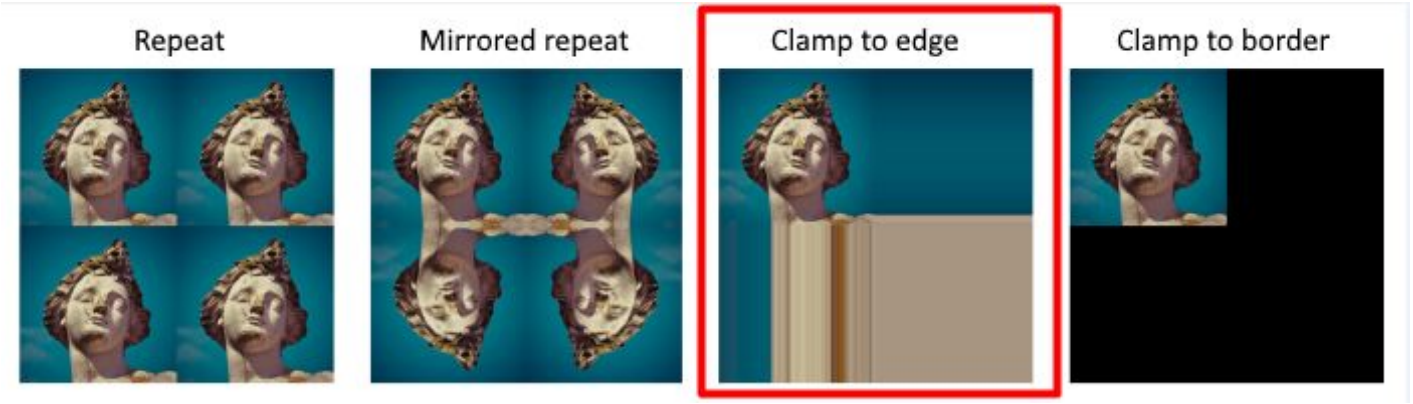
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

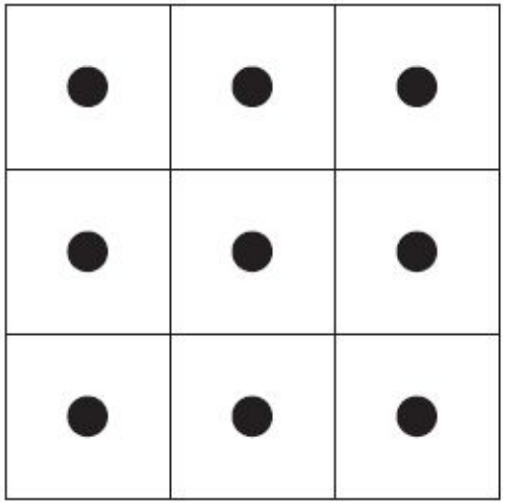
(c)

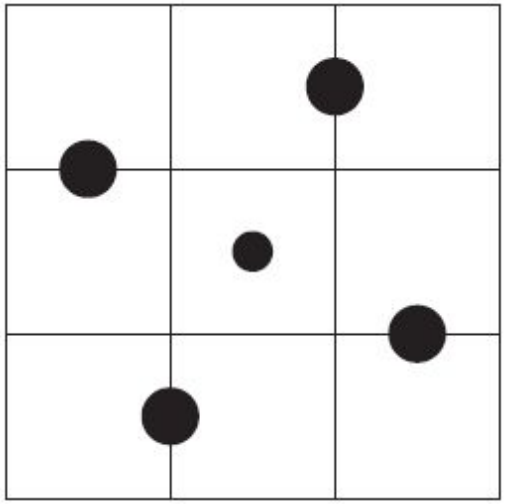
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

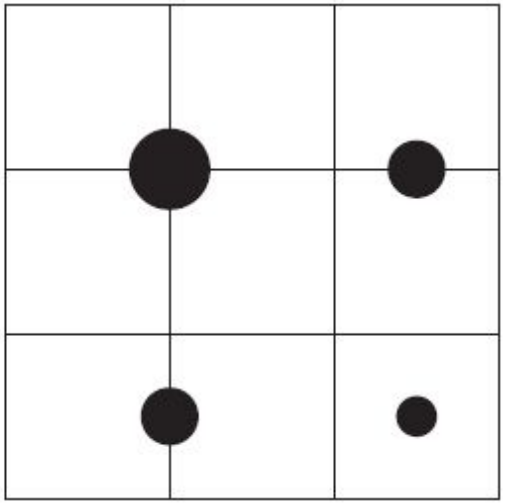
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

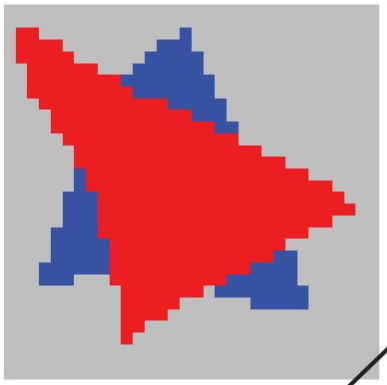
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

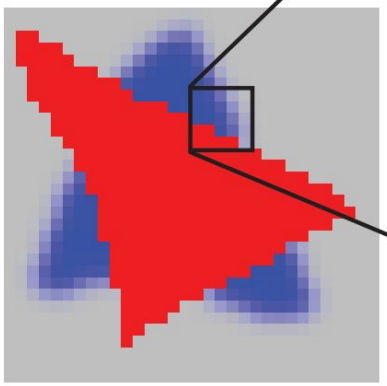
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

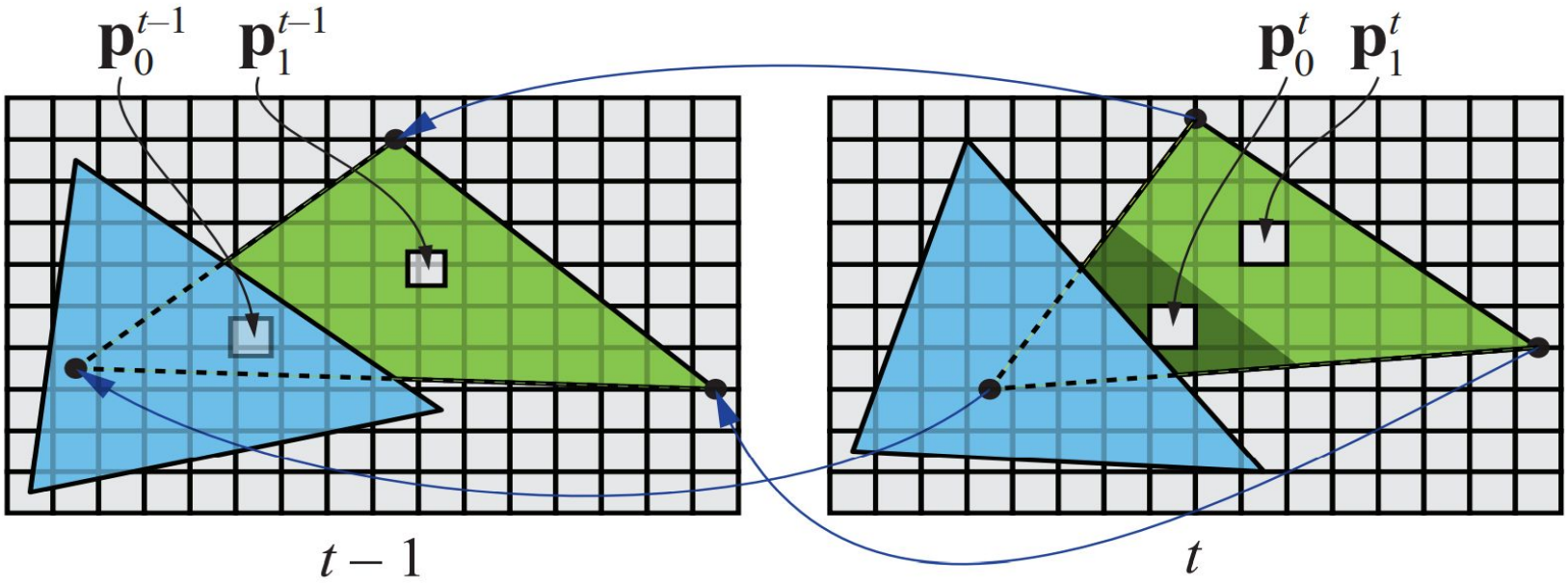
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

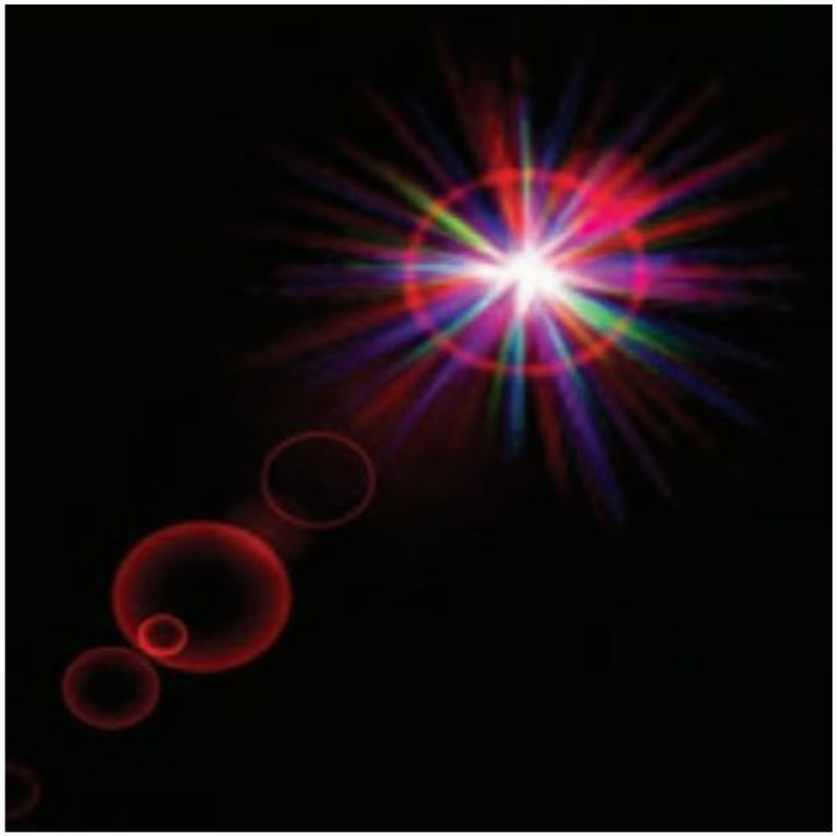
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

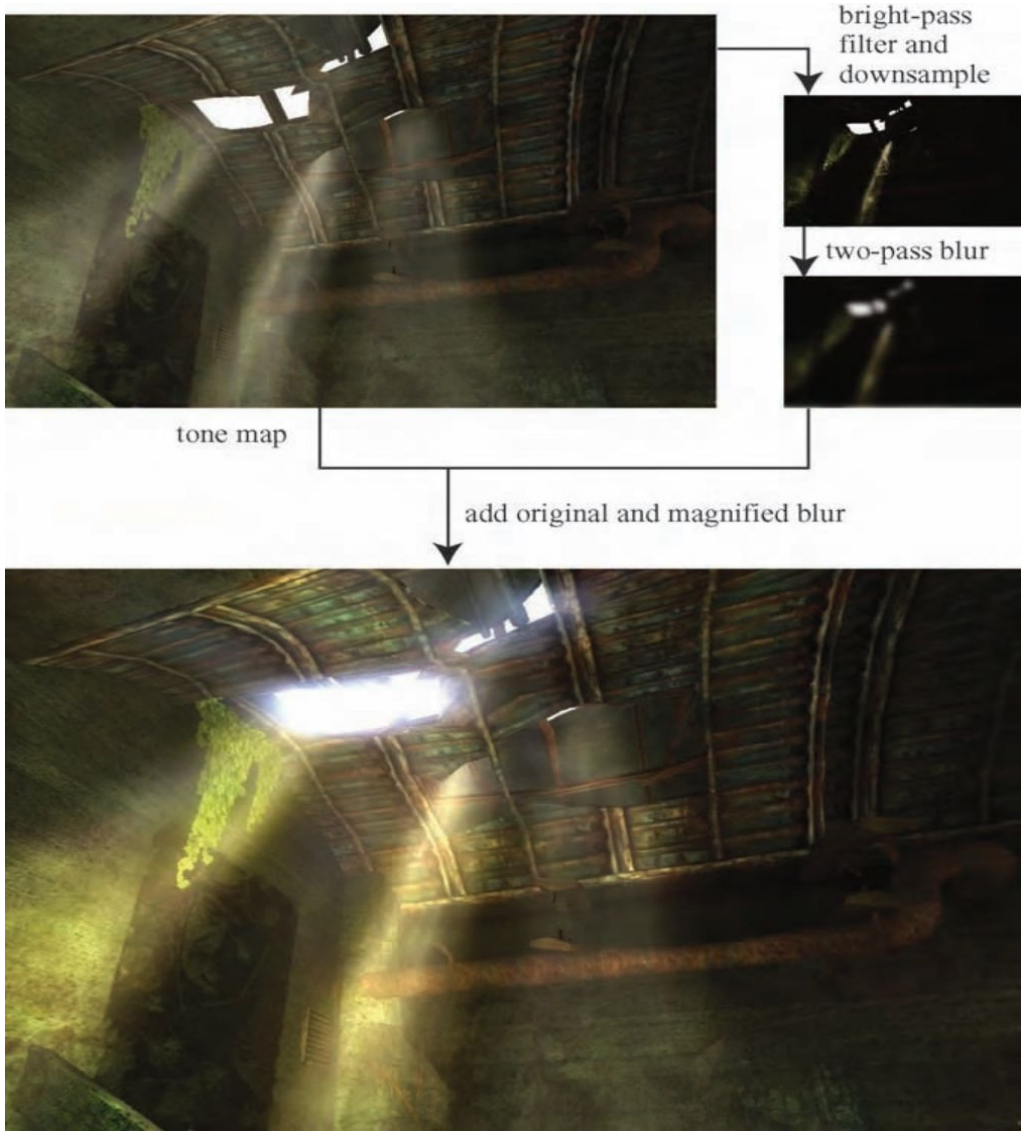
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

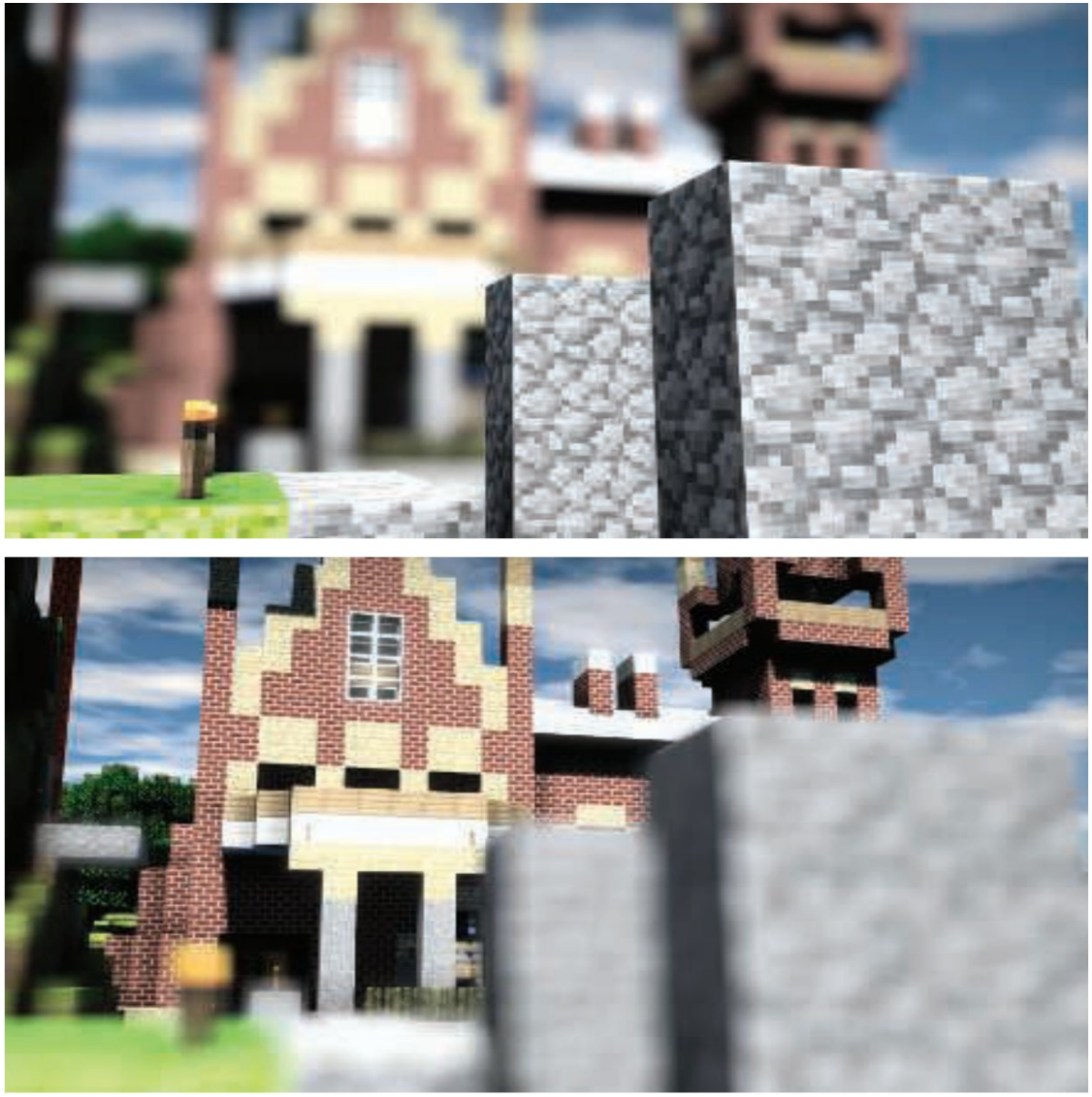
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

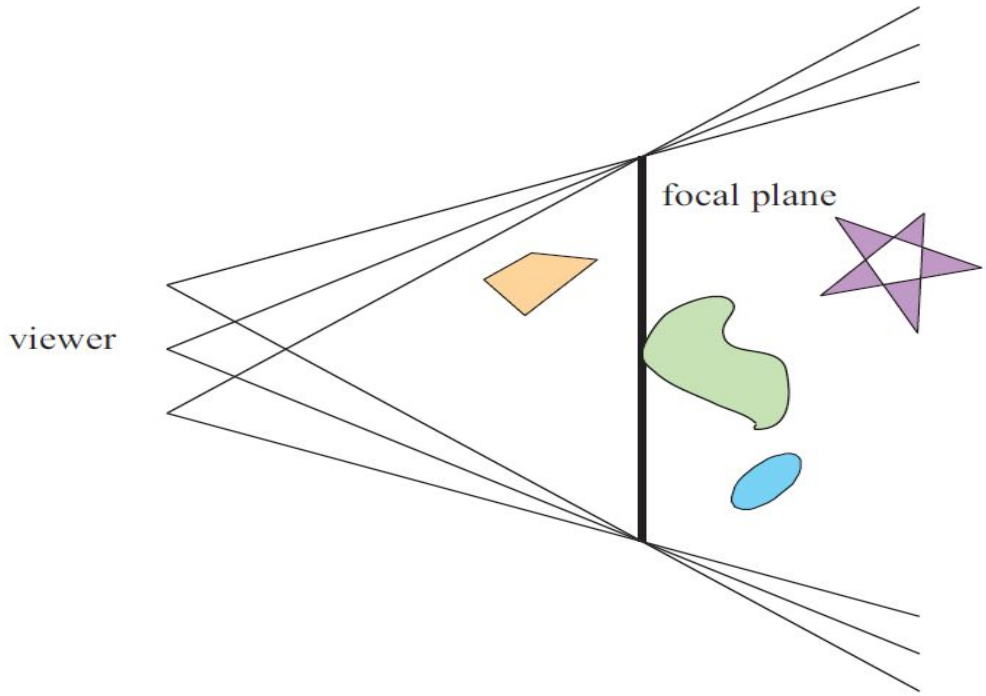


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

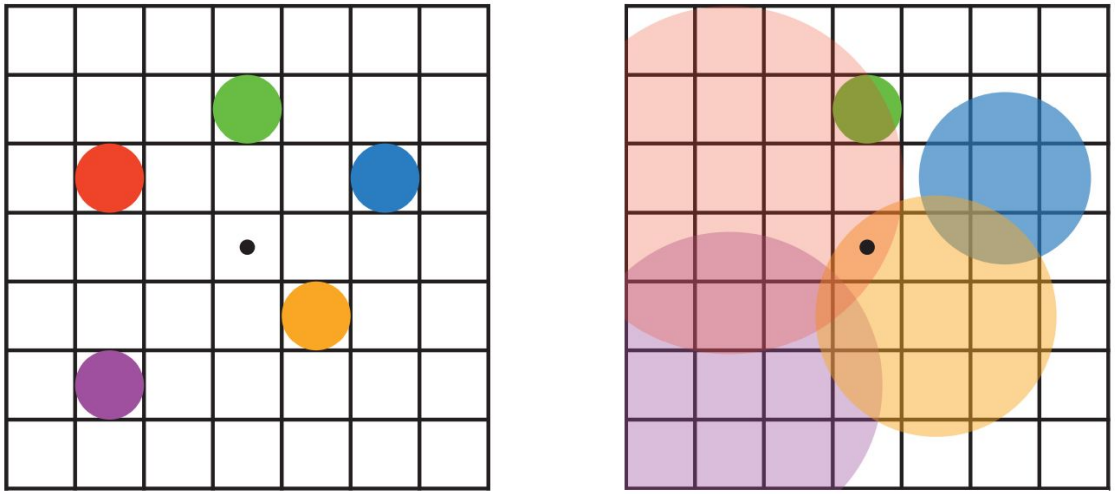
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

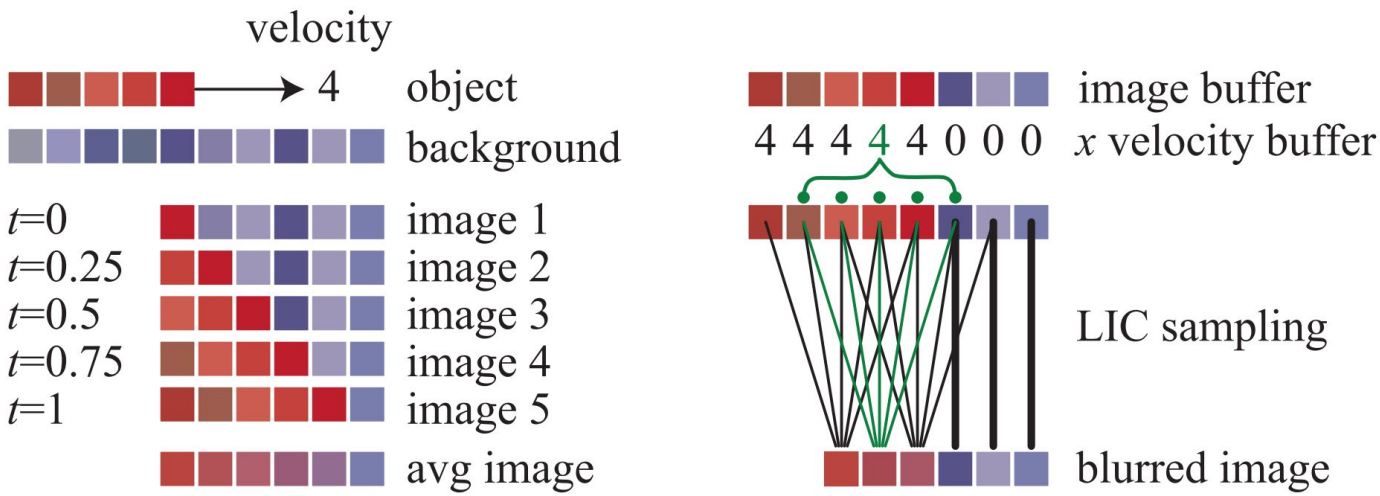
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

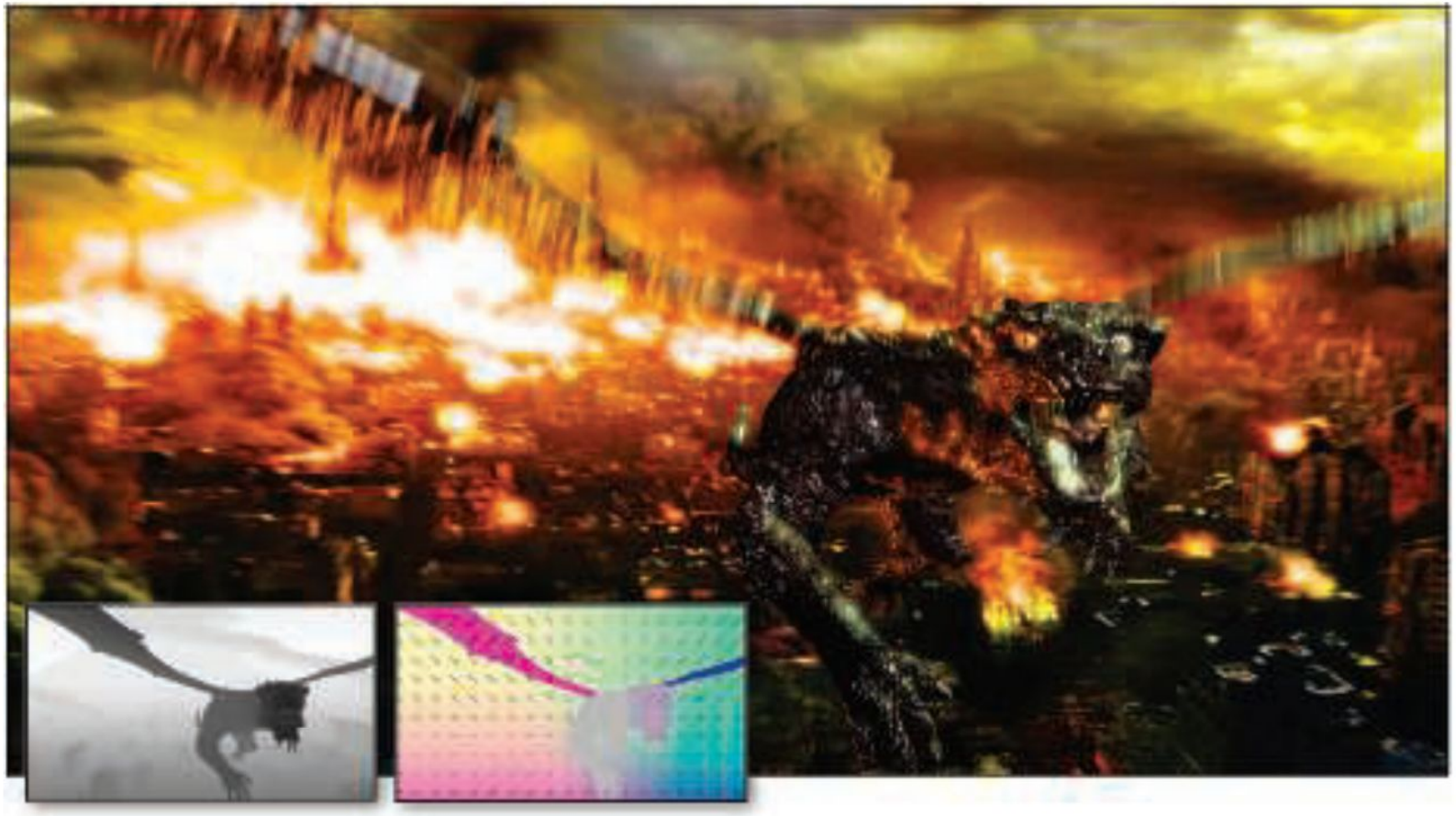
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

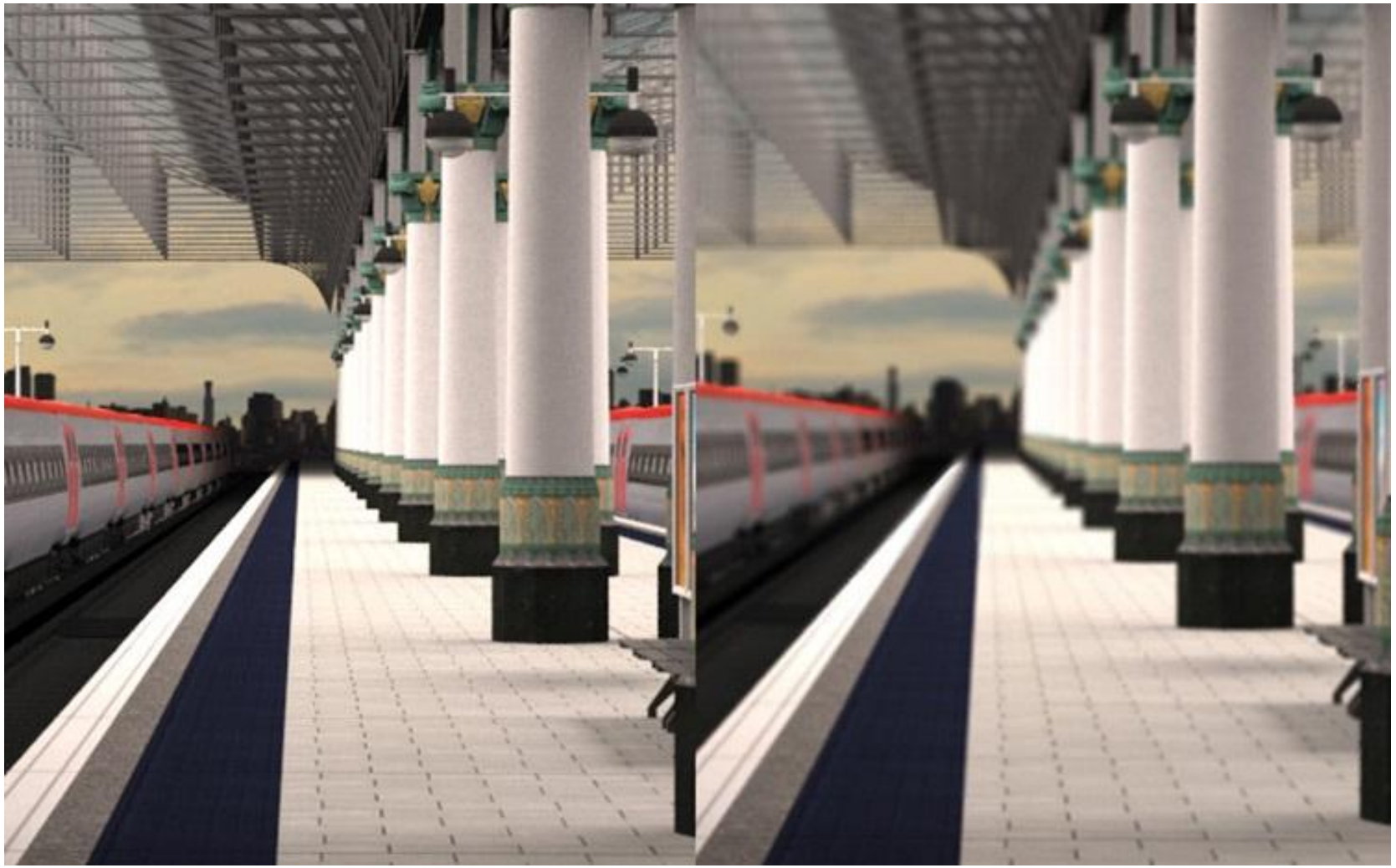
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

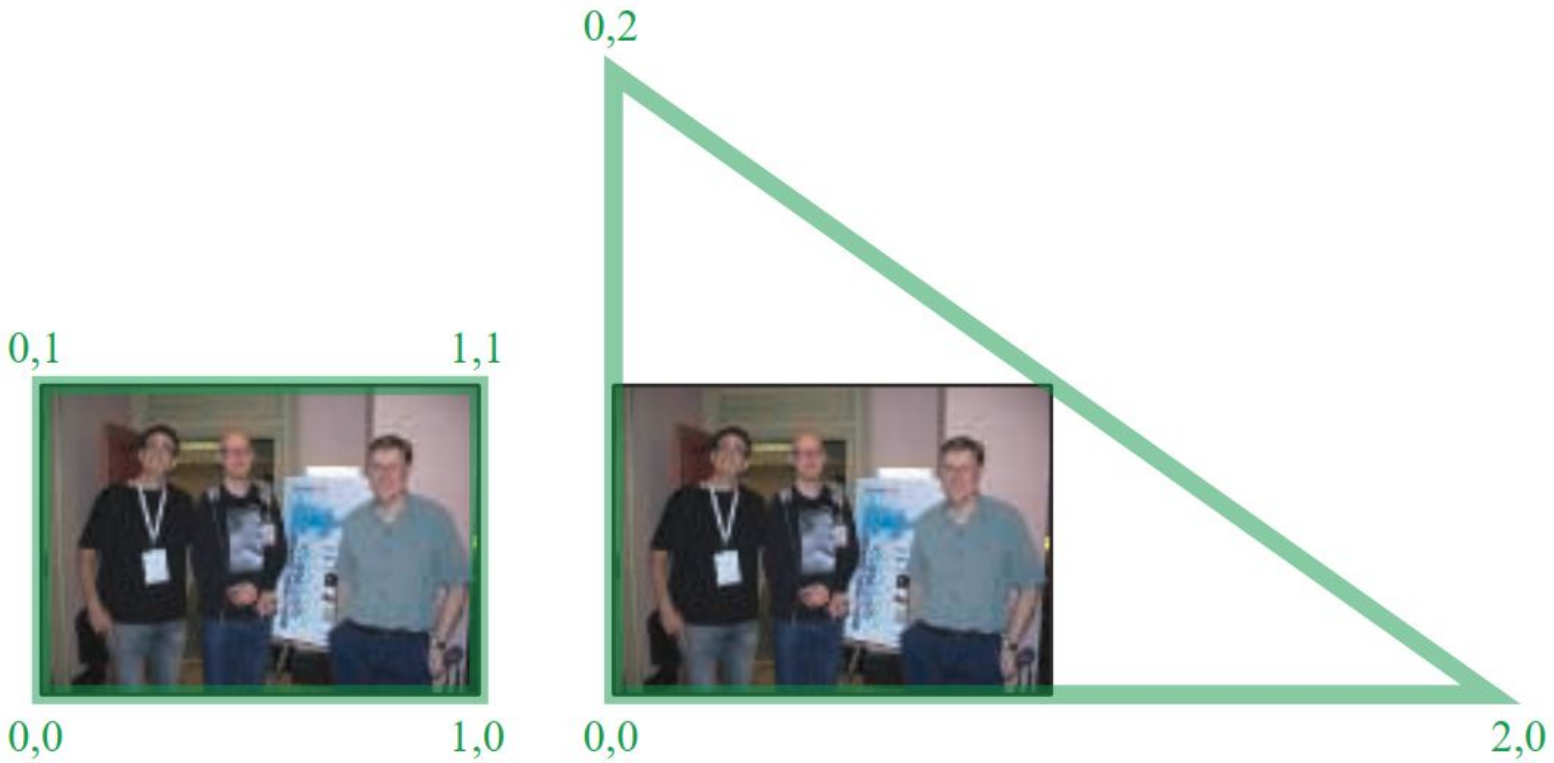
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.
Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

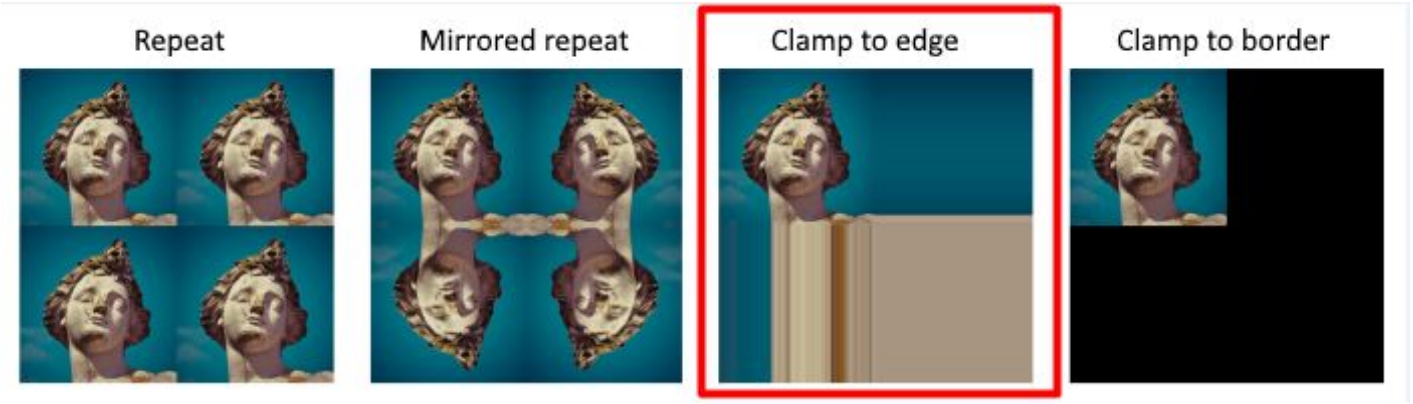
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

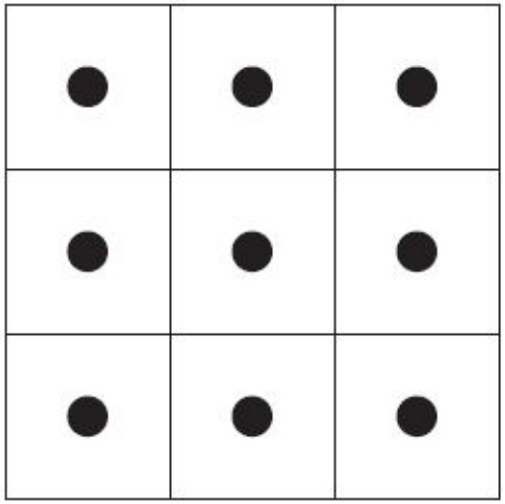
(c)

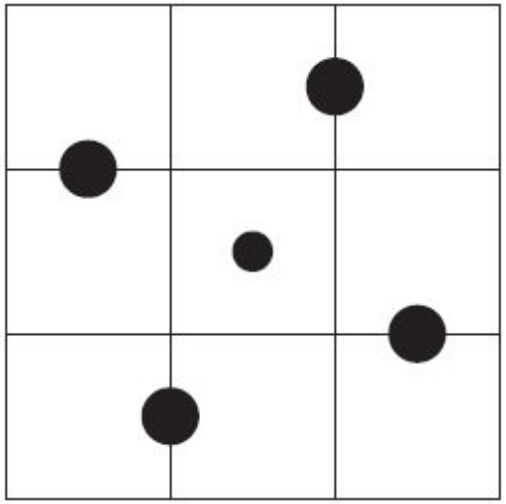
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

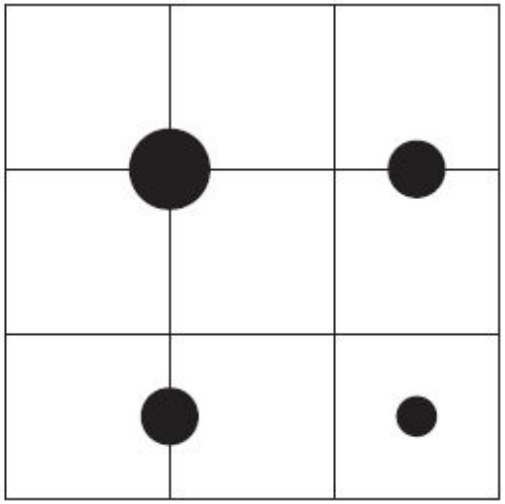
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

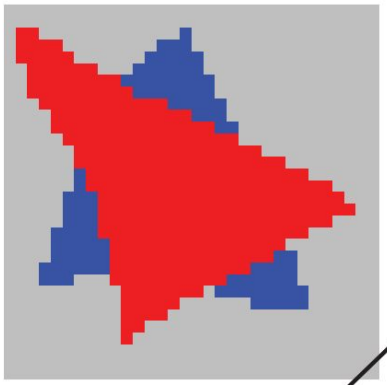
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

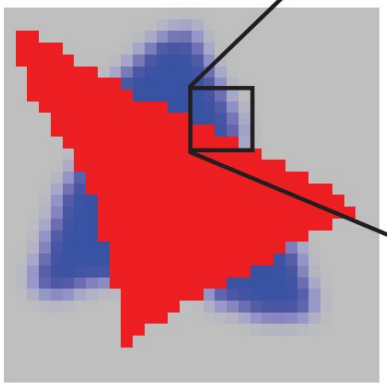
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

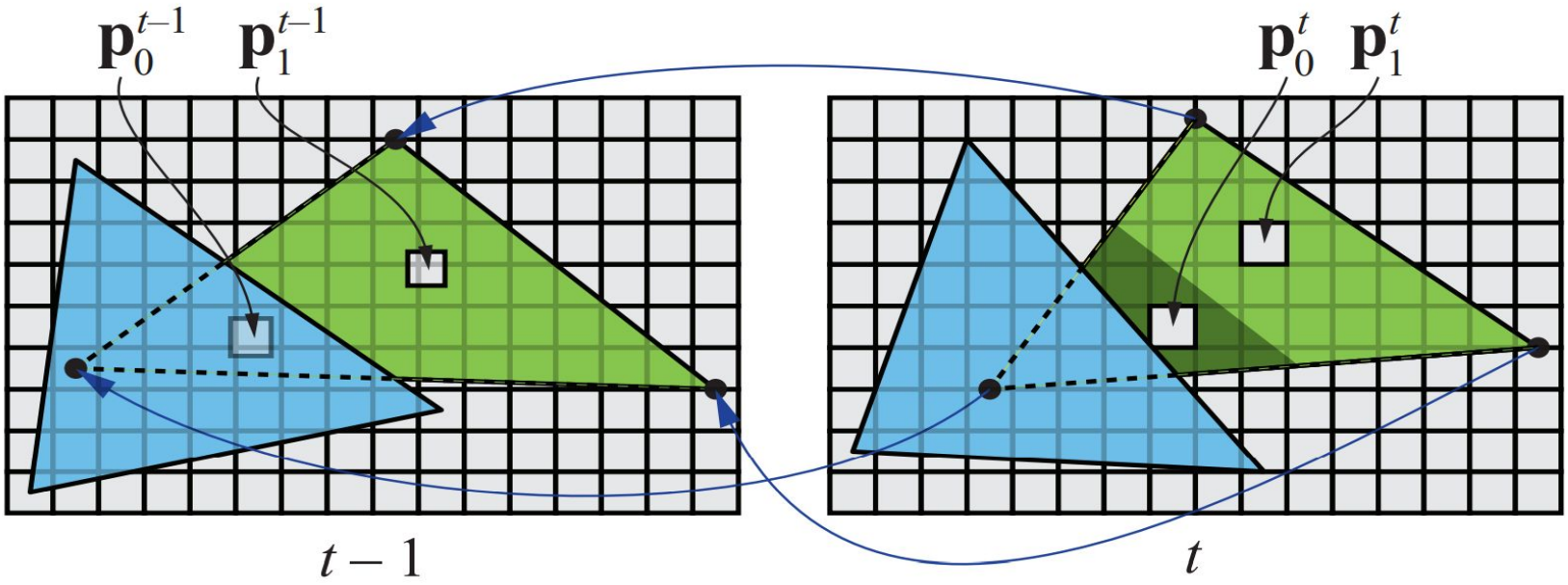
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

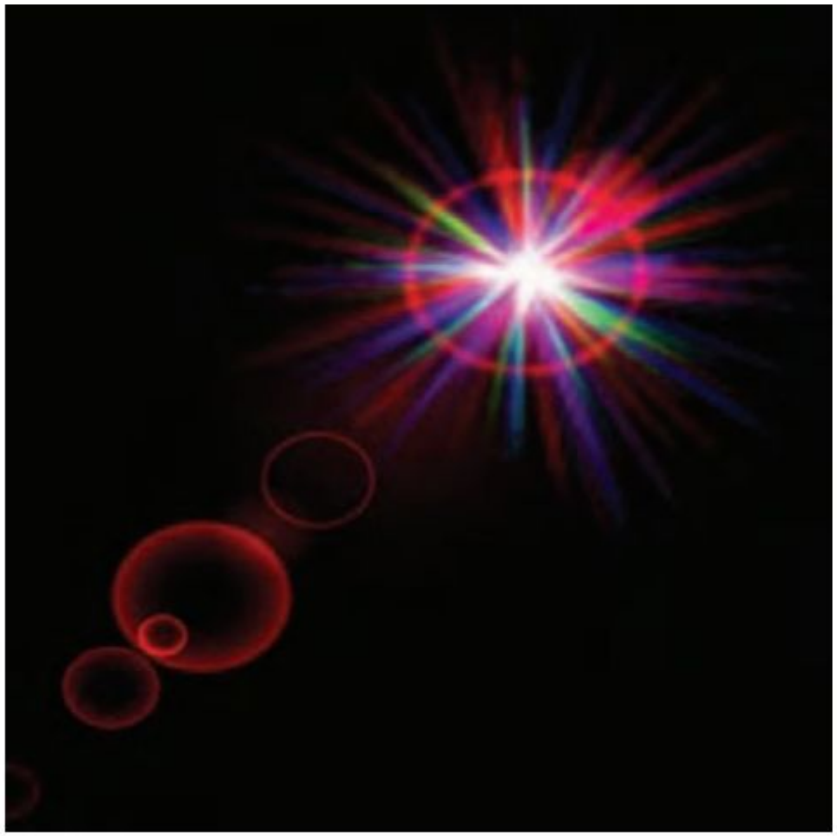
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

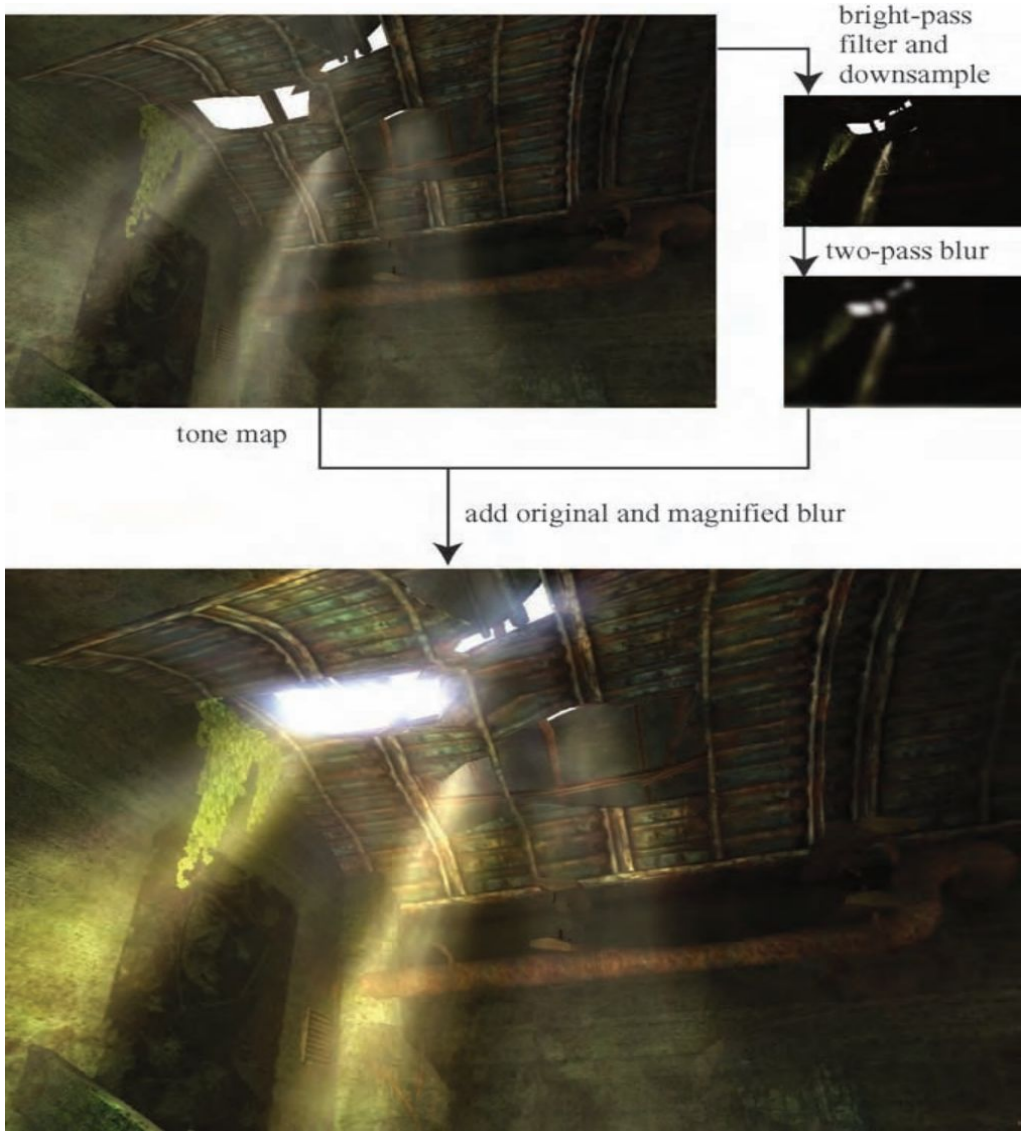
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

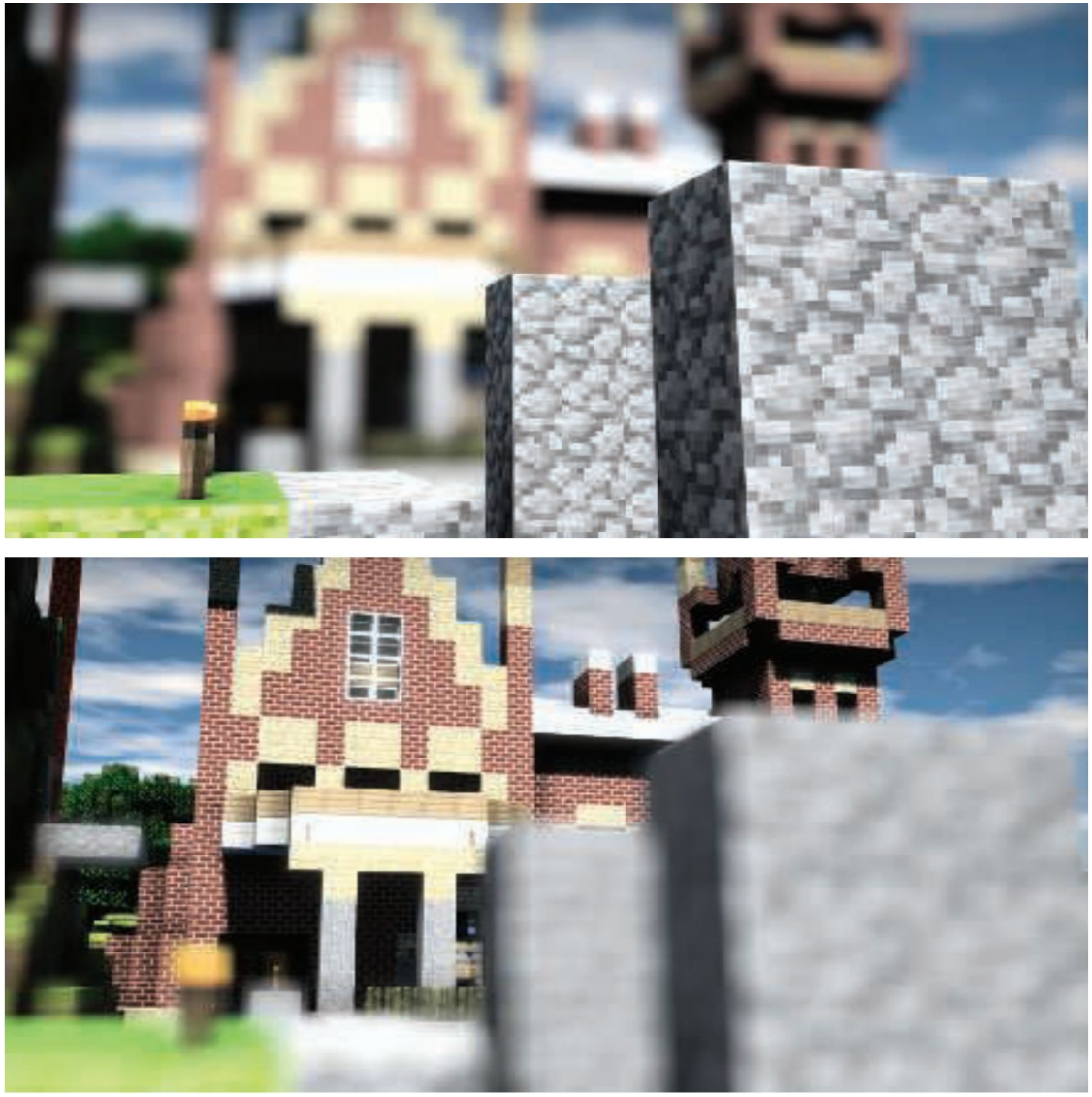
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

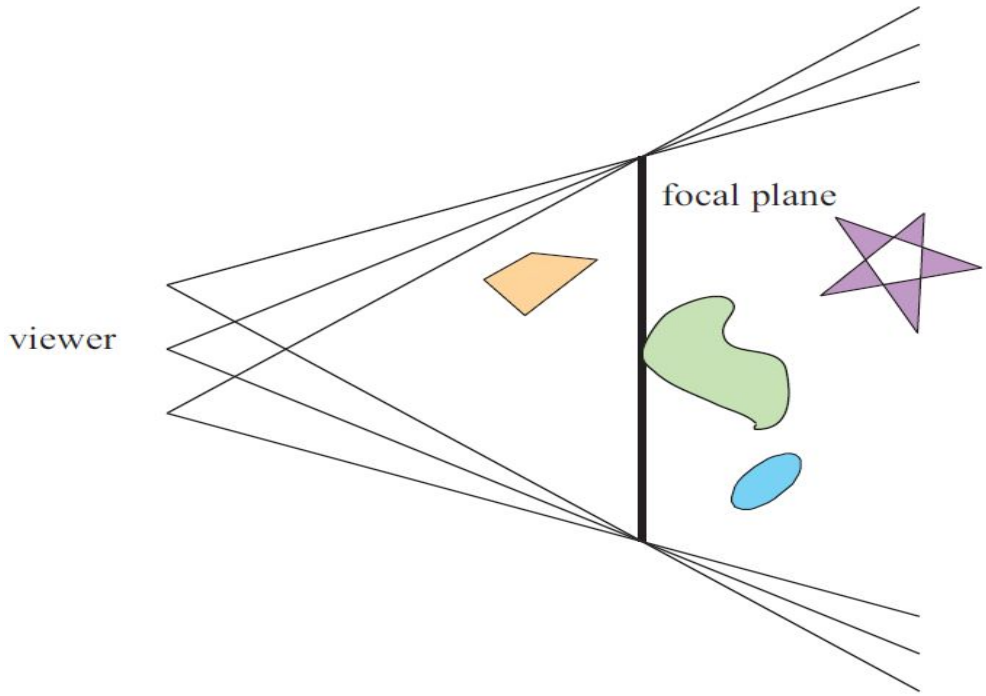


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

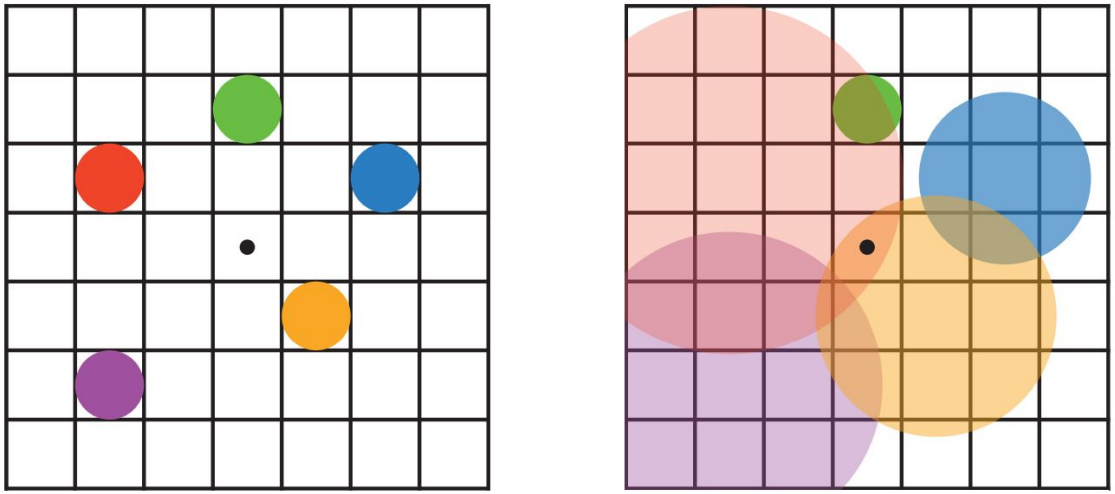
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

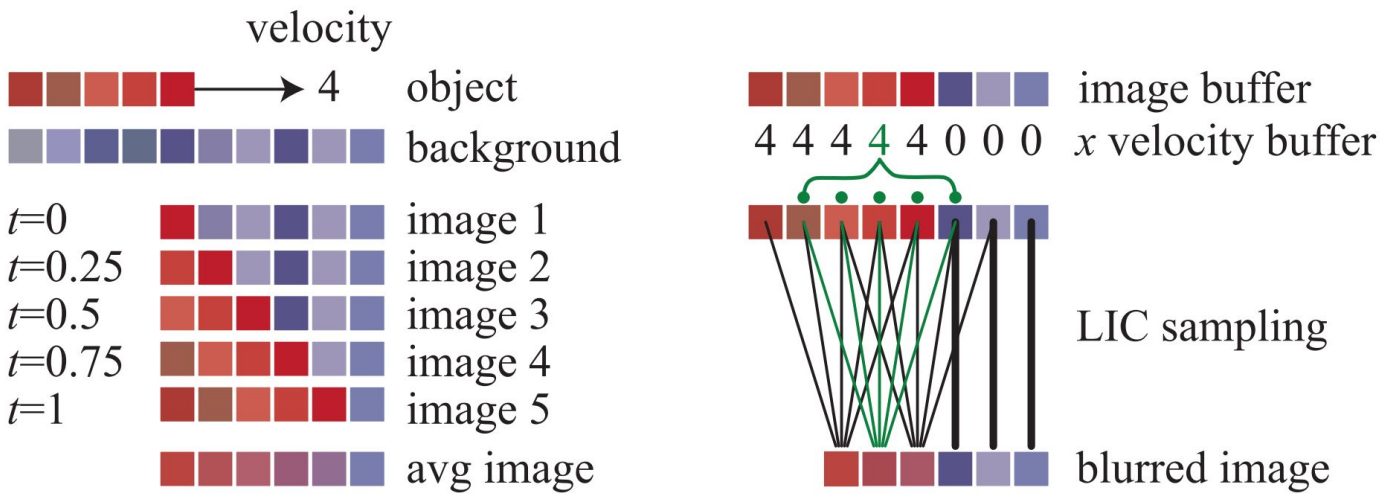
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

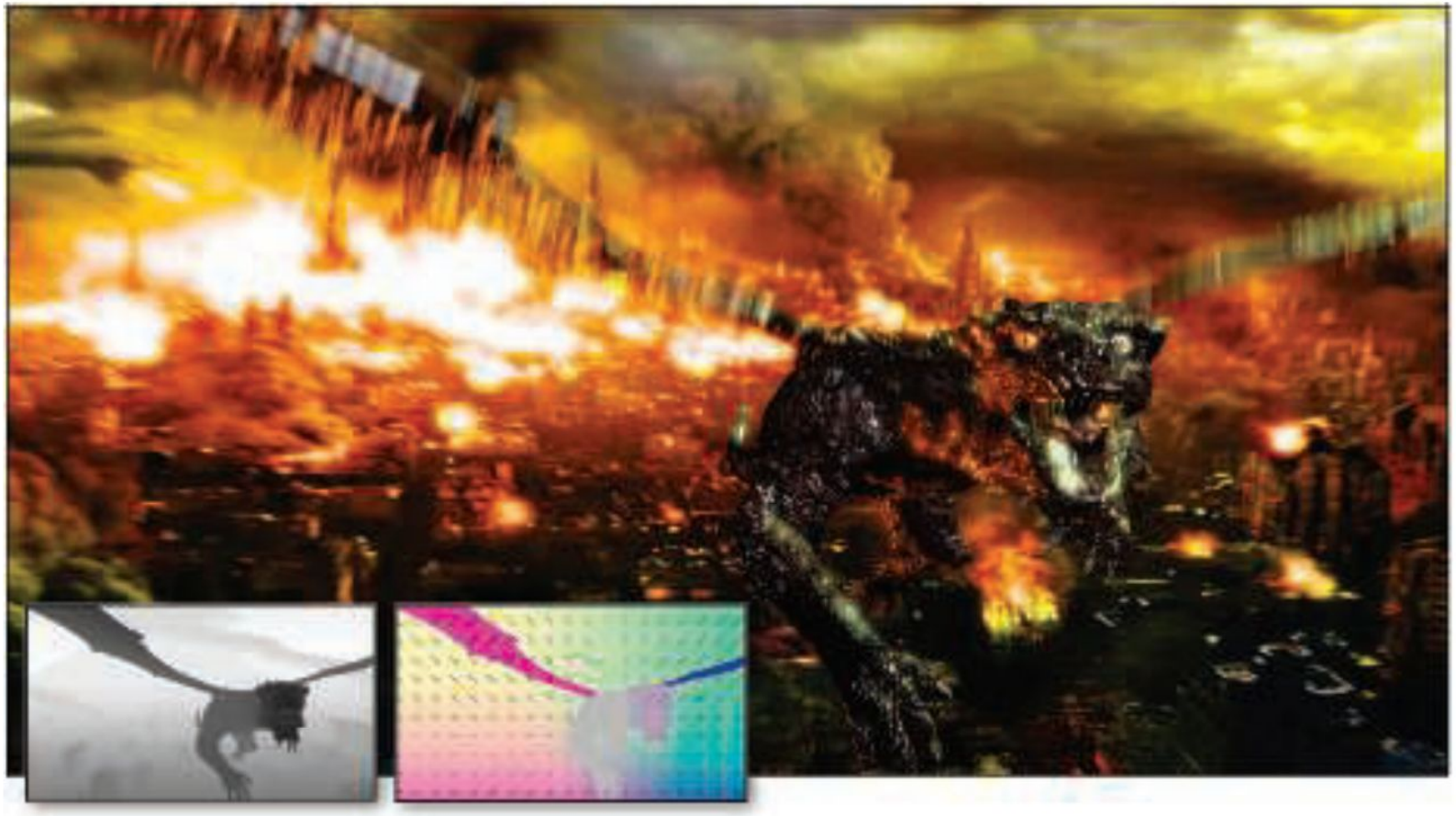
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Efectos basados en imágenes

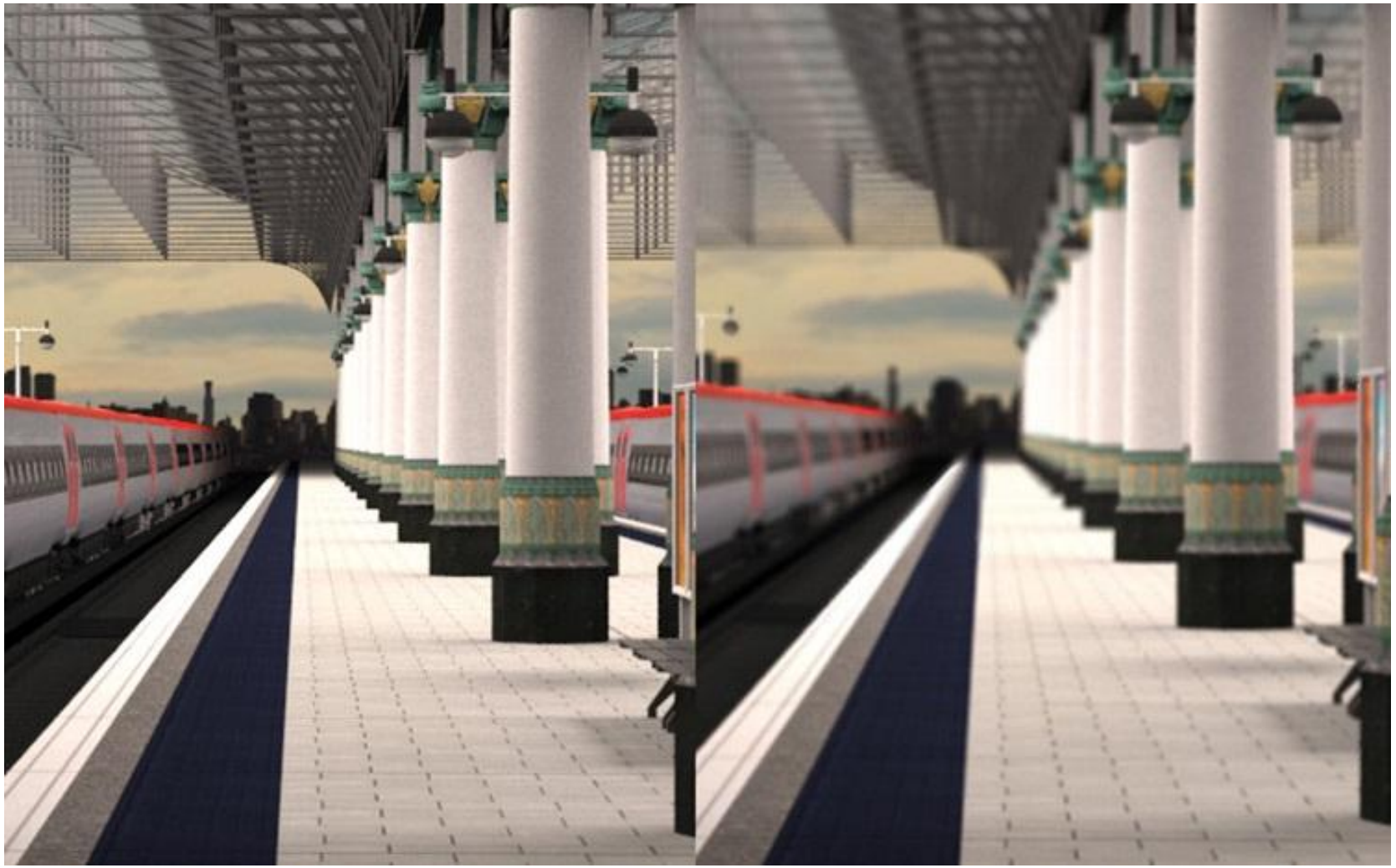
## RTR4 - Capítulo 12

Computación Gráfica Avanzada

Ingeniería en Computación
Facultad de Ingeniería – Universidad de la República
Damián Madeira

# Introducción

Una imagen es más que simplemente retratar objetos



# Temas principales

- Procesamiento de imágenes.
- Técnicas de reproyección.
- Lens Flare y Bloom.
- Depth of field
- Motion blur

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

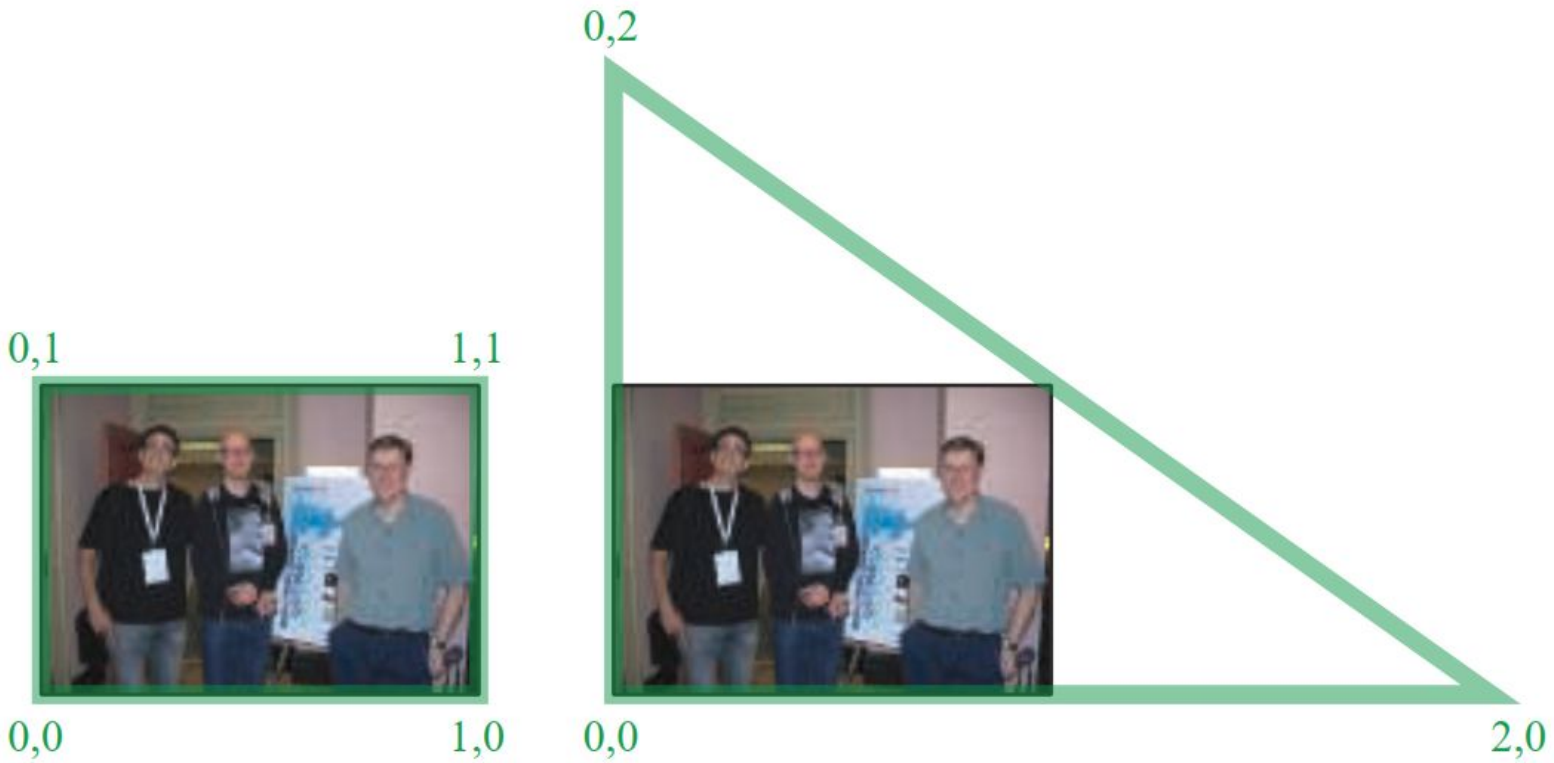
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.
Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

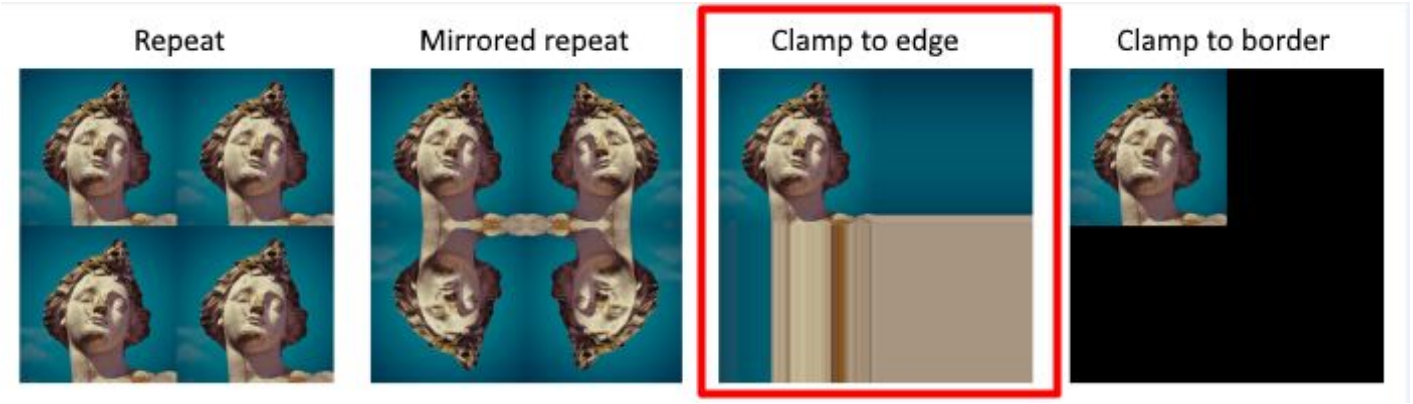
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

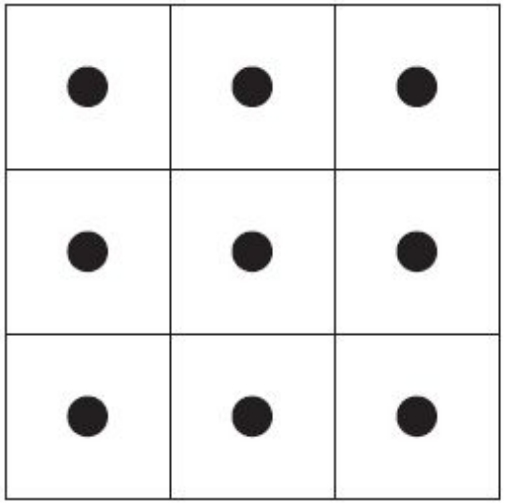
(c)

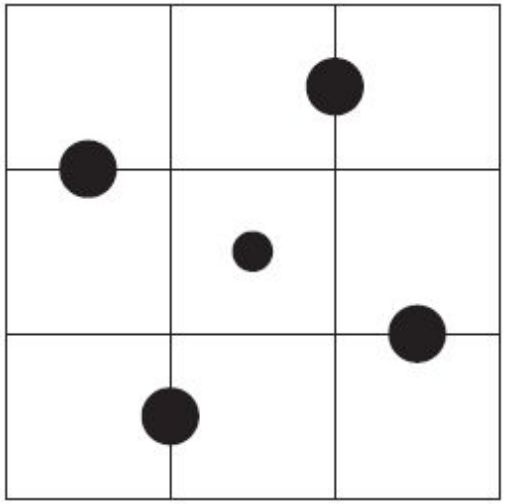
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

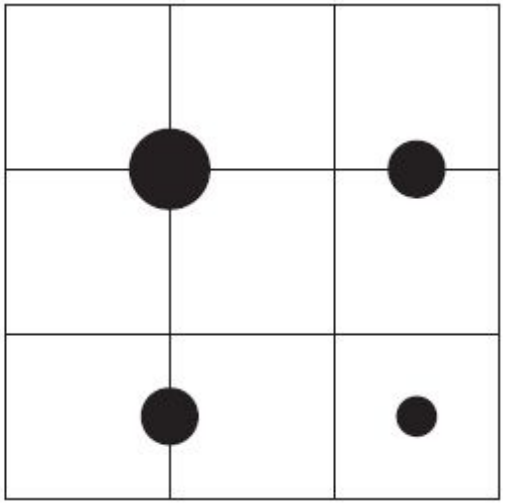
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

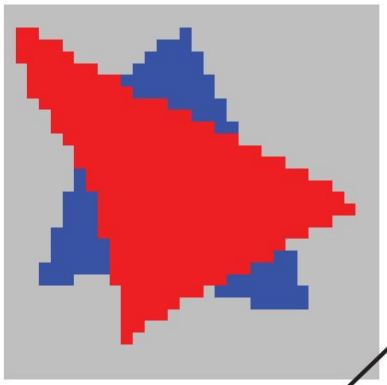
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

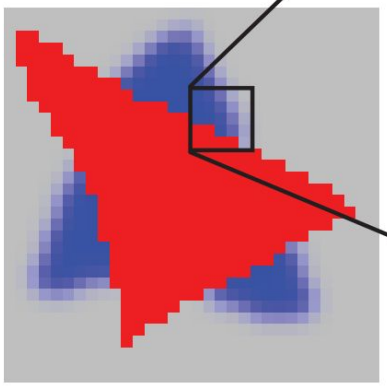
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

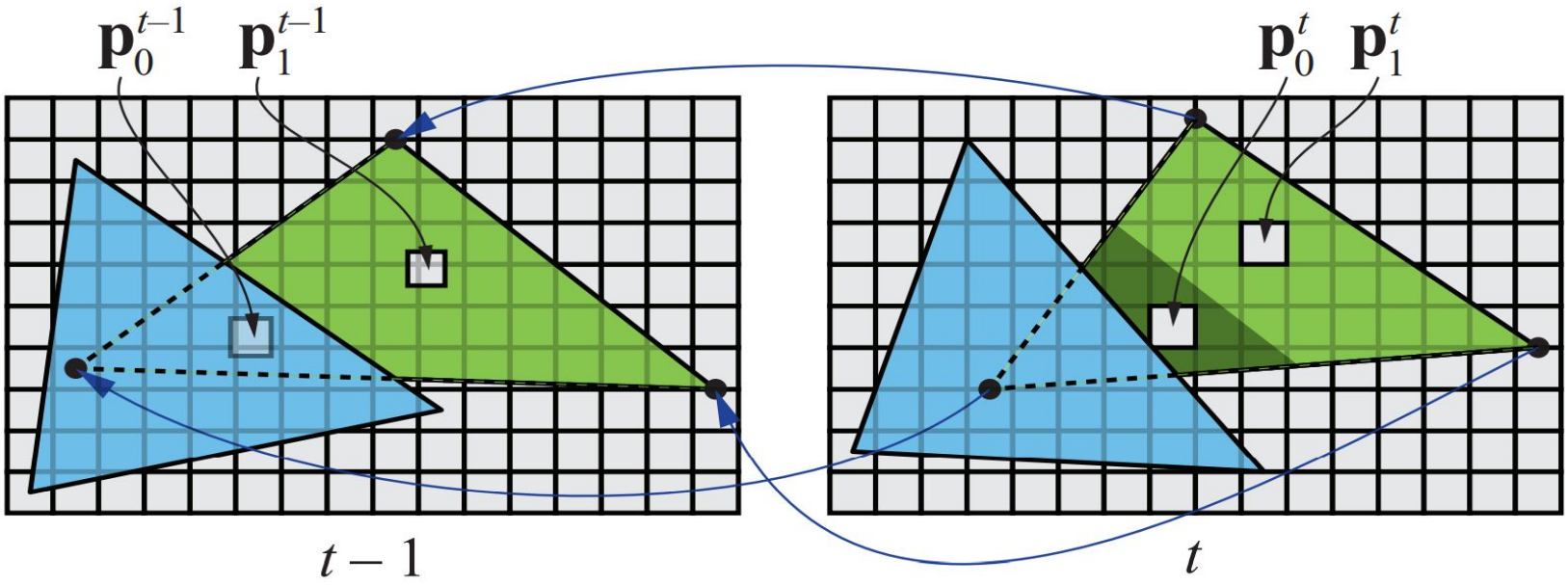
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

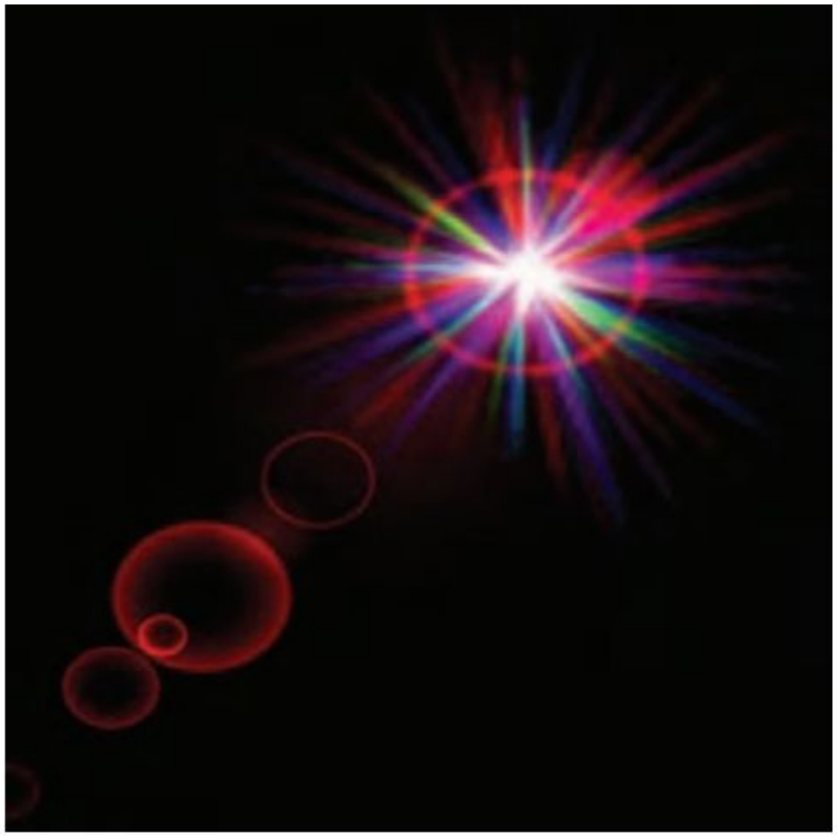
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

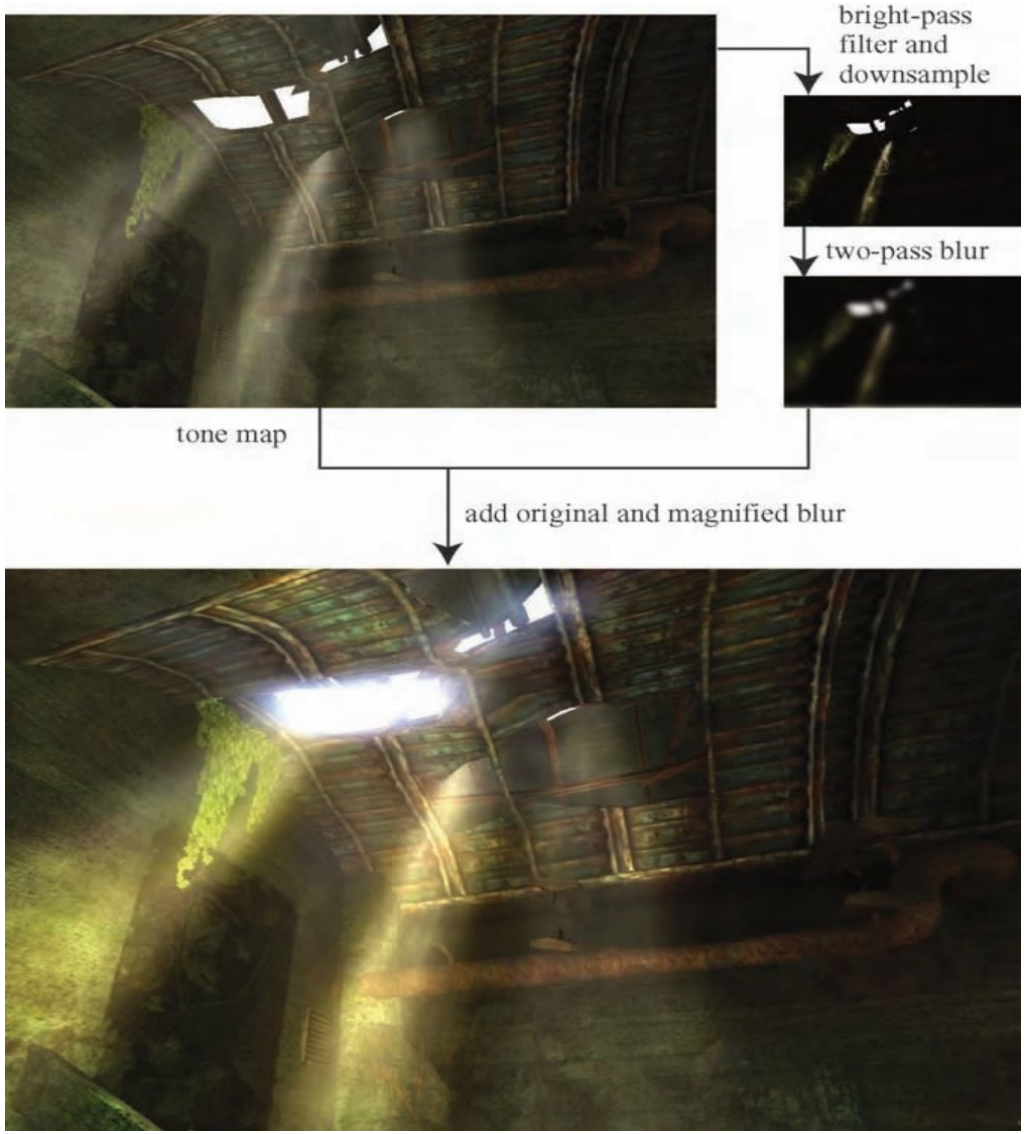
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

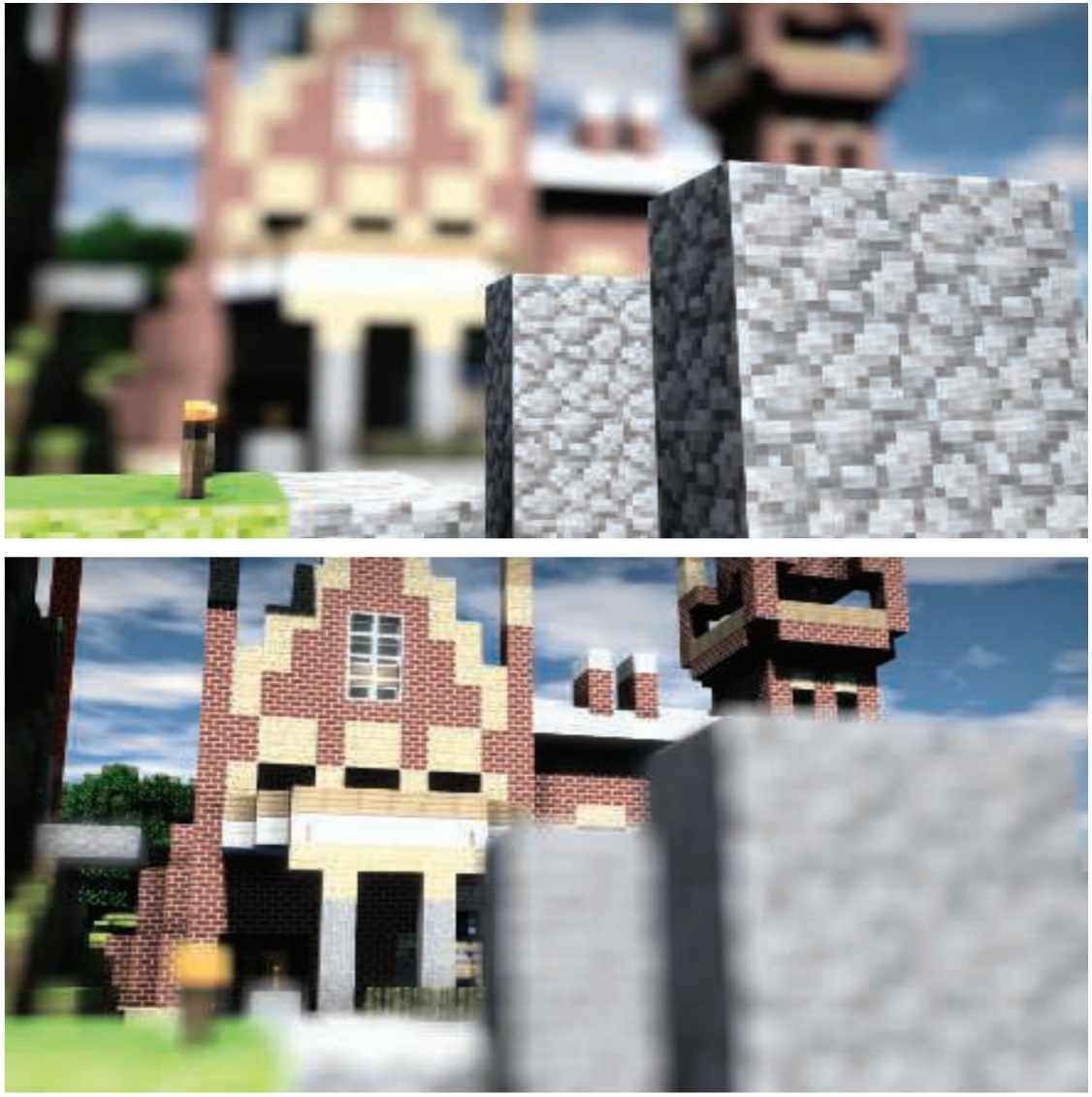
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

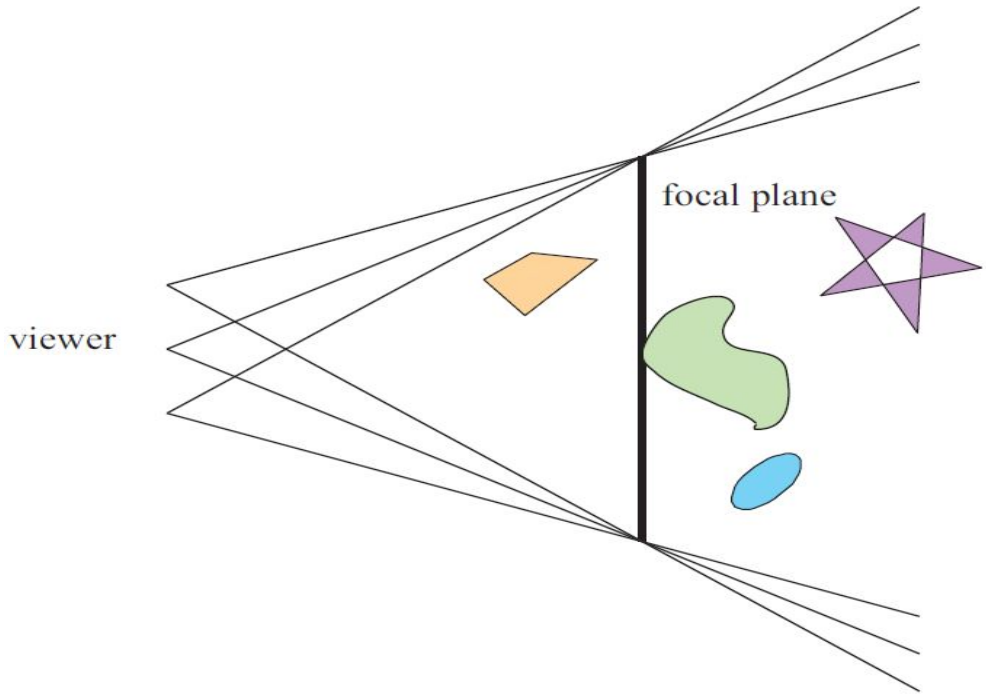


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

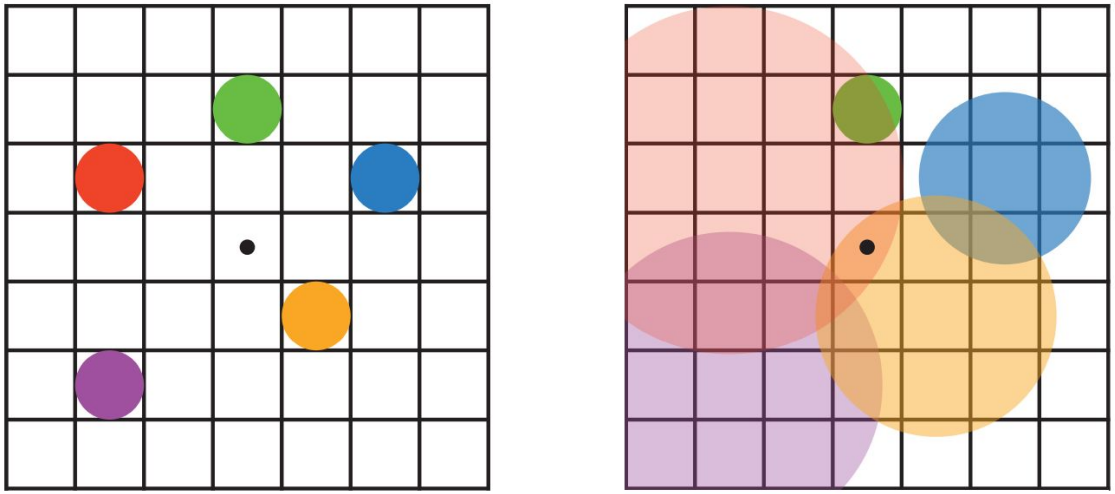
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

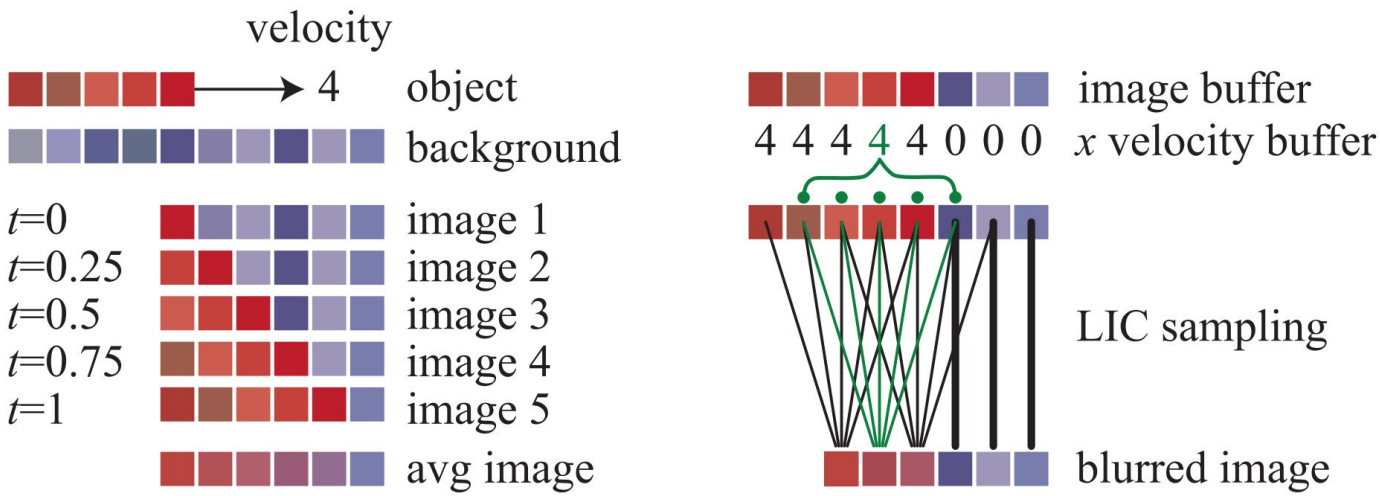
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

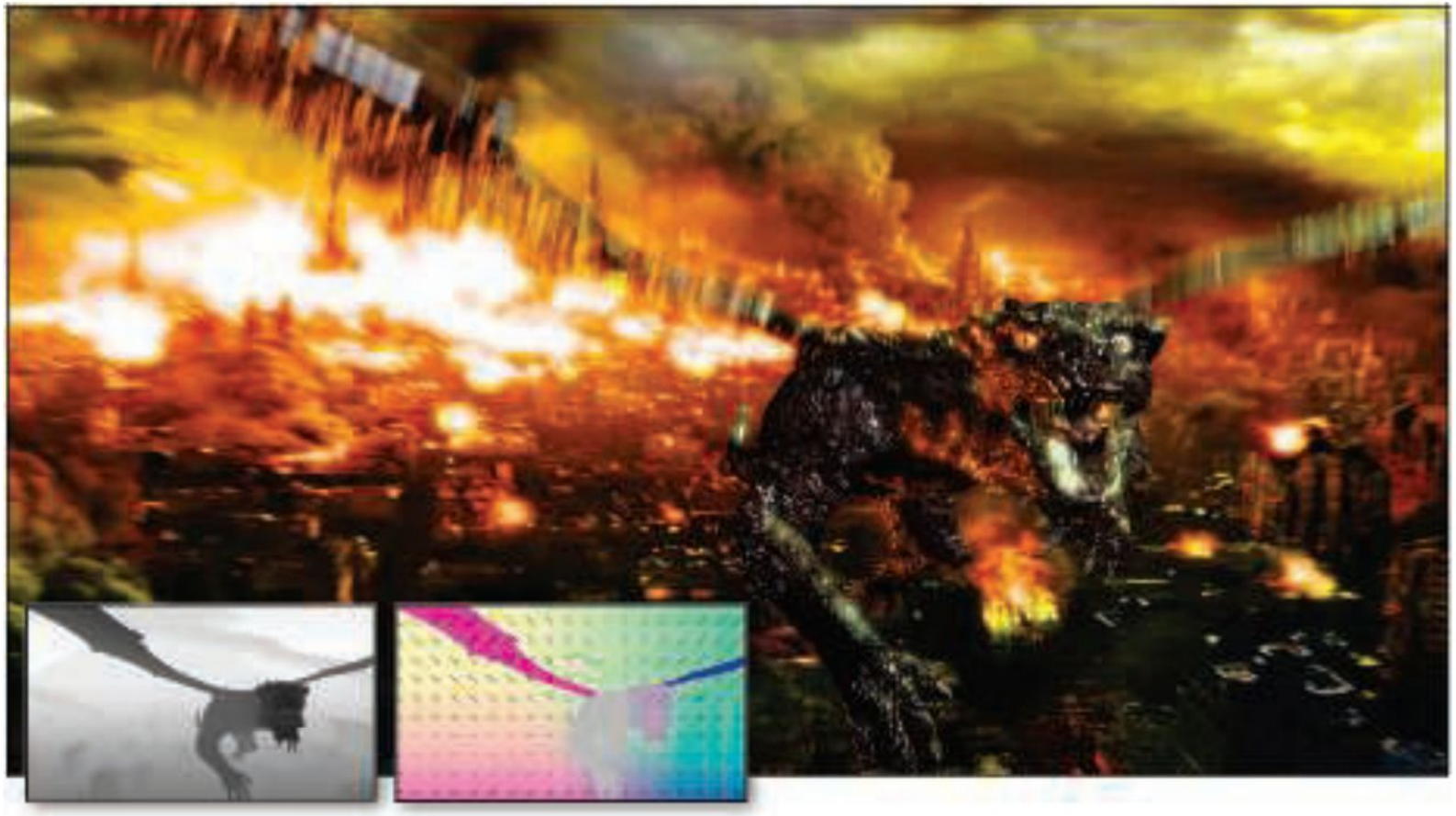
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN

# Procesamiento de imágenes

# Procesamiento de imágenes

- Proceso que toma como entrada una imagen ya renderizada y la analiza y modifica de varias formas
- Dicha imagen se trata como una textura, la cual es aplicada a un cuadrilátero del mismo tamaño que la pantalla
- Este proceso hace uso de los pixel shaders
- El postprocesamiento se realiza renderizando el cuadrilátero, ya que el programa del pixel shader se invocará para cada píxel.

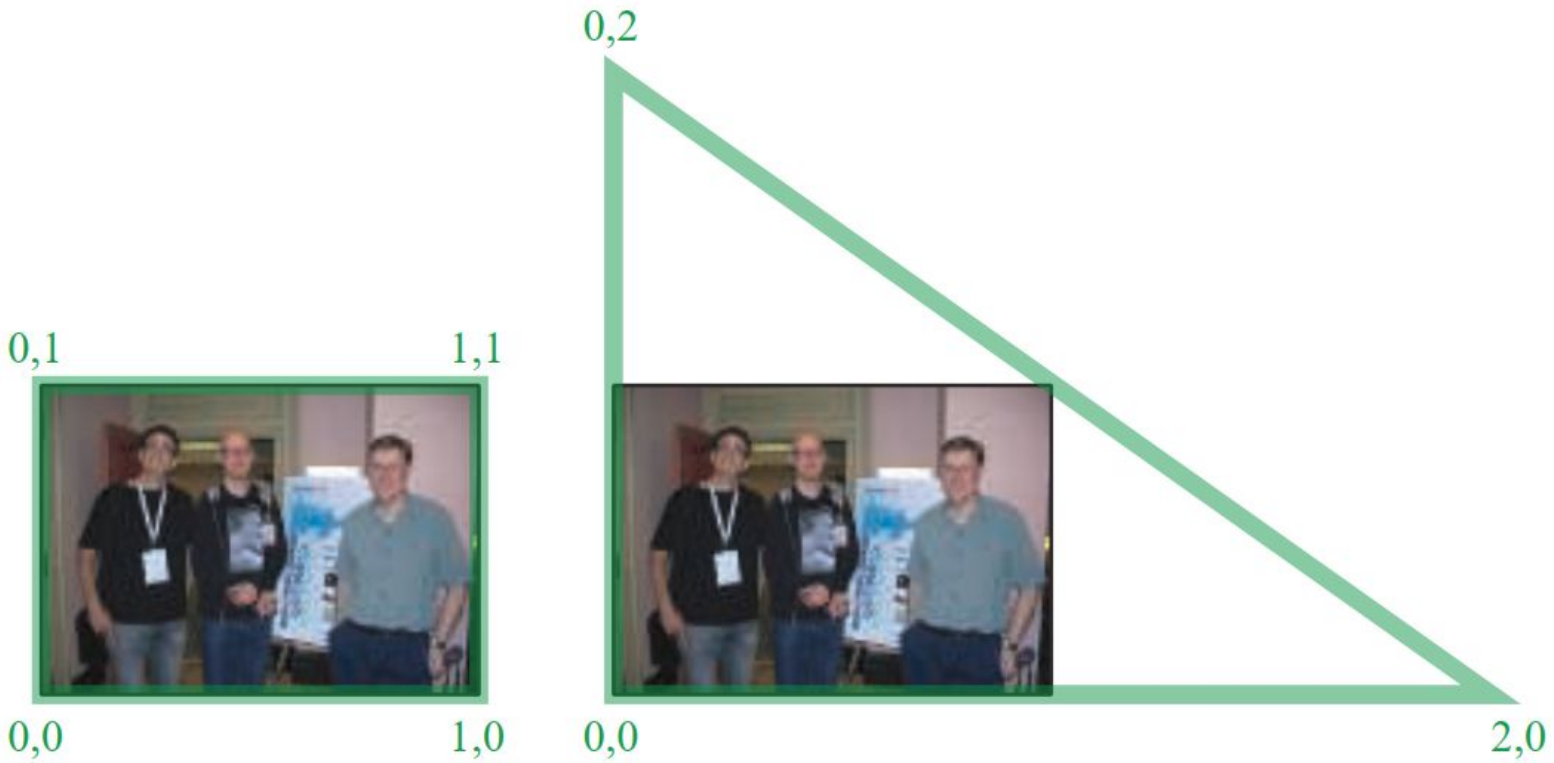
# Procesamiento de imágenes

- La mayoría de los efectos del procesamiento de imágenes se basan en recuperar la información de cada texel de la imagen en el píxel correspondiente.

- Esto se puede hacer asignando coordenadas de textura en el rango [0, 1] al cuadrilátero y escalarlo de acuerdo al tamaño de la imagen de entrada

- En la práctica, en realidad, es más eficiente el uso de un triángulo que contenga la pantalla y no un cuadrilátero formado por dos triángulos

# Procesamiento de imágenes




Según la arquitectura AMD GCN, el procesamiento de imágenes con un único triángulo se realiza un 10% más rápido que con un cuadrilátero.

Esto se debe a que se tiene una mejor coherencia de la caché

# Procesamiento de imágenes

Filter Kernel:

- Es una matriz de convolución utilizada para procesamiento de imágenes.
- Convolución es el proceso de agregar cada elemento de la imagen a sus vecinos locales, ponderados por el kernel.
- La expresión general de una convolución es:

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$$

Donde $g(x,y)$ es la imagen filtrada, $f(x,y)$ la imagen original y $\omega$ es el filter kernel. Se consideran todos los elementos del filter kernel debido a que $-a \leq dx \leq a$ y $-b \leq dy \leq b$

# Procesamiento de imágenes

| | | |
|---|---|---|
| Edge Detection | $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ |  |
| Sharpen | $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ |  |
| Box blur | $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ |  |
| Gaussian blur 3x3 | $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ |  |
| Gaussian blur 5x5 | $\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$ |  |

# Procesamiento de imágenes

- El filtro Gaussiano, con su conocida forma de campana, es el más comúnmente utilizado para este tipo de procesos.

$$\text{Gaussian}(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) e^{-\frac{r^2}{2\sigma^2}}$$

donde r es la distancia desde el centro del texel y σ es la desviación estándar

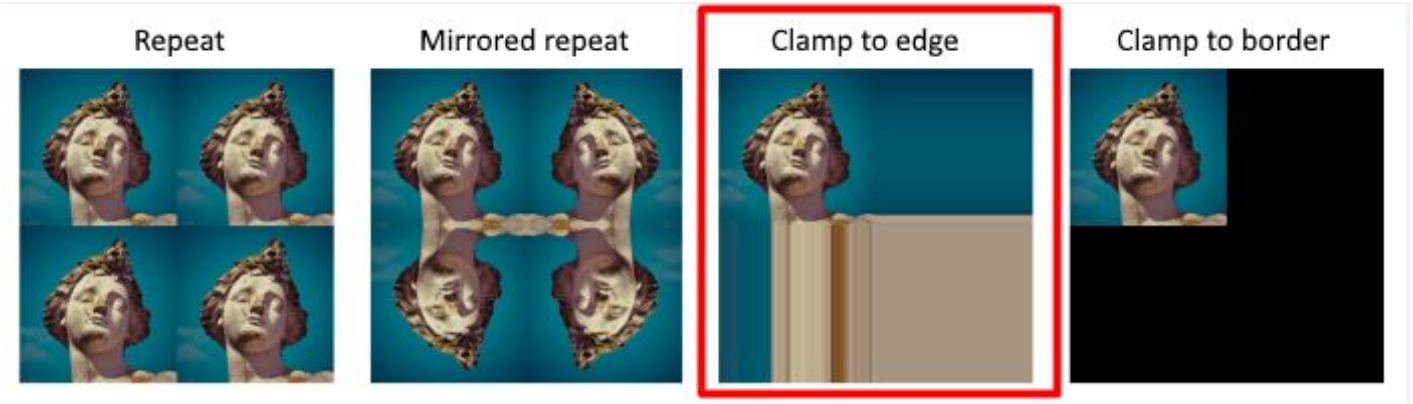
- Dado que cuando se crea el kernel, los pesos computados por texel se suman juntos sobre el área debajo de la curva y luego todos los valores se dividen por esta suma, el parámetro constante de la fórmula de arriba se vuelve irrelevante. Esto sucede porque la suma final de todos estos pesos suma 1 por construcción y por ende es innecesaria la normalización que aporta dicho coeficiente, y por este motivo, la mayoría de las veces ni siquiera aparece este término en estos casos.

# Procesamiento de imágenes

Un problema que surge es que, al tomar muestras en los pixeles de algunas de las esquinas, por ejemplo utilizando muestras de tamaño 3x3, la operación de filtro va a intentar recuperar texels que están fuera de los límites de la imagen

Existen dos formas de solucionar este inconveniente:

- Setear la textura para que sea clamp to the edge
- Renderizar la imagen original a una resolución apenas más grande que la del display para que esos texels fuera de la pantalla existan.



# Procesamiento de imágenes

Uso de un único filtro gaussiano de dos dimensiones (a)

Vs

uso de dos filtros gaussianos de una dimensión realizados en serie (b y c)

(a)

| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |

El costo de acceso a los texels en el caso (a) es de orden $d^2$ mientras que el costo en los casos (b) y (c) es 2d, siendo d el diámetro del kernel

(b)

| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |
| 0.0545 | 0.2442 | 0.4026 | 0.2442 | 0.0545 |

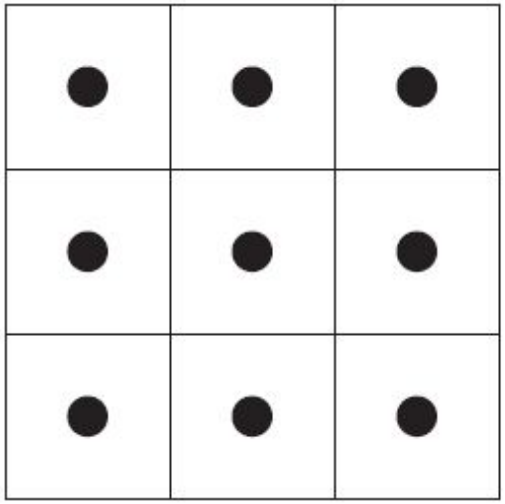
(c)

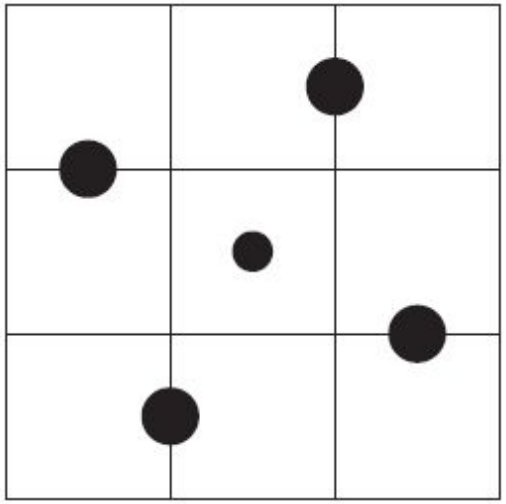
| | | | | |
|---|---|---|---|---|
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.4026 | 0.4026 | 0.4026 | 0.4026 | 0.4026 |
| 0.2442 | 0.2442 | 0.2442 | 0.2442 | 0.2442 |
| 0.0545 | 0.0545 | 0.0545 | 0.0545 | 0.0545 |

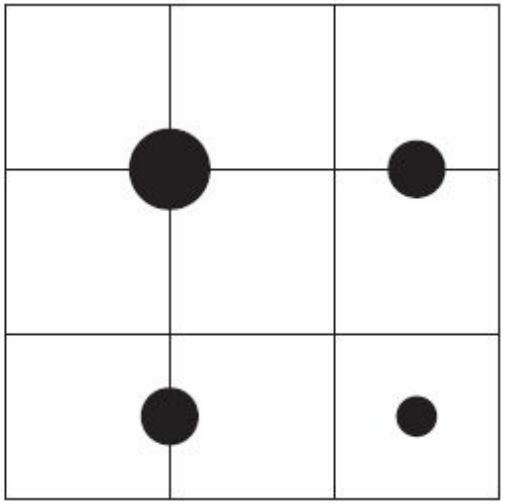
# Procesamiento de imágenes

Por ejemplo, supongamos que el objetivo es usar un box filter, tomar el promedio de los nueve texels que forman una cuadrícula de 3 × 3 alrededor de un texel dado y mostrar este resultado borroso

Existen diferentes formas de abordarlo:







Más eficiente, reduce accesos a textura

# Procesamiento de imágenes

**Downsampling:**

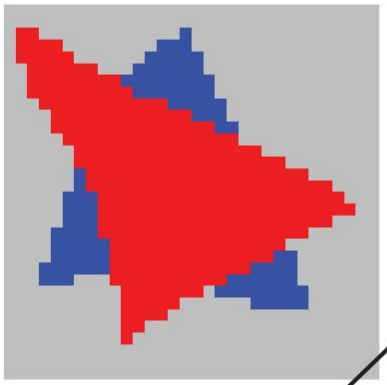
Es un técnica bastante utilizada en filtros de blurring. Consiste en disminuir la resolución de la imagen original, por ejemplo, dividiendo a la mitad los tamaños en ambos ejes, lo cual deriva en una imagen de tamaño ¼ de la original. Luego, cuando se accede a esta imagen para mezclar en la imagen final de resolución completa, se amplía la textura utilizando interpolación bilineal para mezclar las muestras.

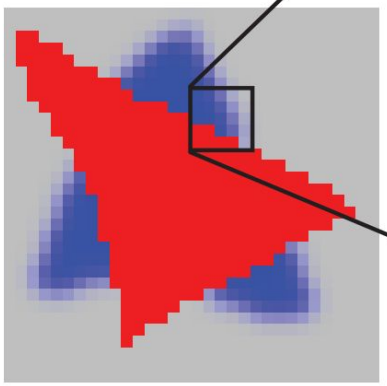
Esto provoca un efecto de blur que, si bien tiene calidad inferior a lo que sería el mismo filtro aplicado a una imagen en su resolución original, es bastante útil cuando se necesita aplicar blur en grandes zonas de color similar.

Además, al dividir la resolución de la pantalla, se necesitan muchos menos accesos a texels, lo cual lo vuelve un método bastante eficiente

# Procesamiento de imágenes

**Bilateral Filter:** Es un filtro cuyo principal objetivo es descartar o reducir la influencia de las muestras que parecen no estar relacionadas con la superficie en la muestra central que se está evaluando





| | | | | |
|---|---|---|---|---|
| 0.0030 | 0.0133 | 0.0219 | 0.0133 | 0.0030 |
| 0.0133 | 0.0596 | 0.0983 | 0.0596 | 0.0133 |
| 0.0219 | 0.0983 | 0.1621 | 0.0983 | 0.0219 |
| | | 0.0983 | 0.0596 | 0.0133 |
| | | | | 0.0030 |

# Procesamiento de imágenes

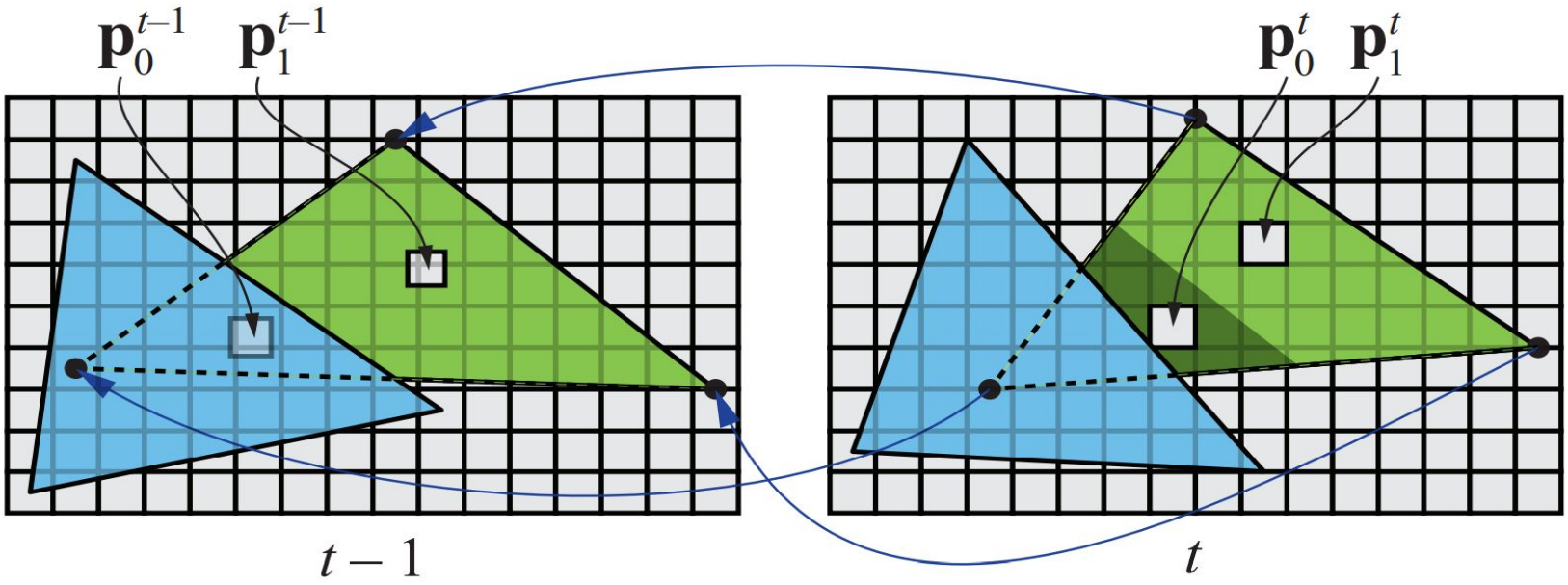
# Técnicas de Reproyección

# Técnicas de Reproyección

La reproyección se basa en la idea de reutilizar muestras que fueron computadas en frames anteriores. Como su nombre lo implica, estas muestras se reutilizan, en la medida que sea posible, cuando varía el punto de vista y/o la orientación con respecto a frames anteriores.

El objetivo principal que persigue es la reducción del costo general de renderizado a lo largo de varios frames.

Existen dos tipos: reverse reprojection y forward reprojection



# Técnicas de Reproyección

Si bien esta técnica es muy útil para disminuir el costo de renderizado, debido a que la reutilización de los shaded values supone que son independientes de cualquier tipo de movimiento, no es conveniente reutilizar los shaded values durante muchos frames.
Para asegurarse de que esto no suceda, existen dos formas que son las más usadas:

- Que se realice un refresco automático de los valores cada algunos frames.

  Para esto se sugiere dividir la pantalla en *n* grupos, donde cada grupo es una selección pseudo-random de regiones de 2x2 pixeles, y que en cada frame se actualice uno de los grupos.
- Un filtro llamado *running-average filter* que gradualmente va descartando los valores viejos.

  El filtro se describe como:

$$\mathbf{c}_f(\mathbf{p}^t) = \alpha \mathbf{c}(\mathbf{p}^t) + (1-\alpha)\mathbf{c}(\mathbf{p}^{t-1})$$

# Lens Flare y Bloom

# Lens Flare y Bloom

- Lens Flare o destello de lente, es un fenómeno causado por la luz cuando esta viaja a través de un sistema de lentes por reflexión indirecta. Un ejemplo de estos son los halos de luz.

- Por otro lado, el fenómeno Bloom o resplandor es causado por la dispersión en la lente y otras partes del ojo, creando un brillo alrededor de la luz y atenuando el contraste en otras partes la escena.

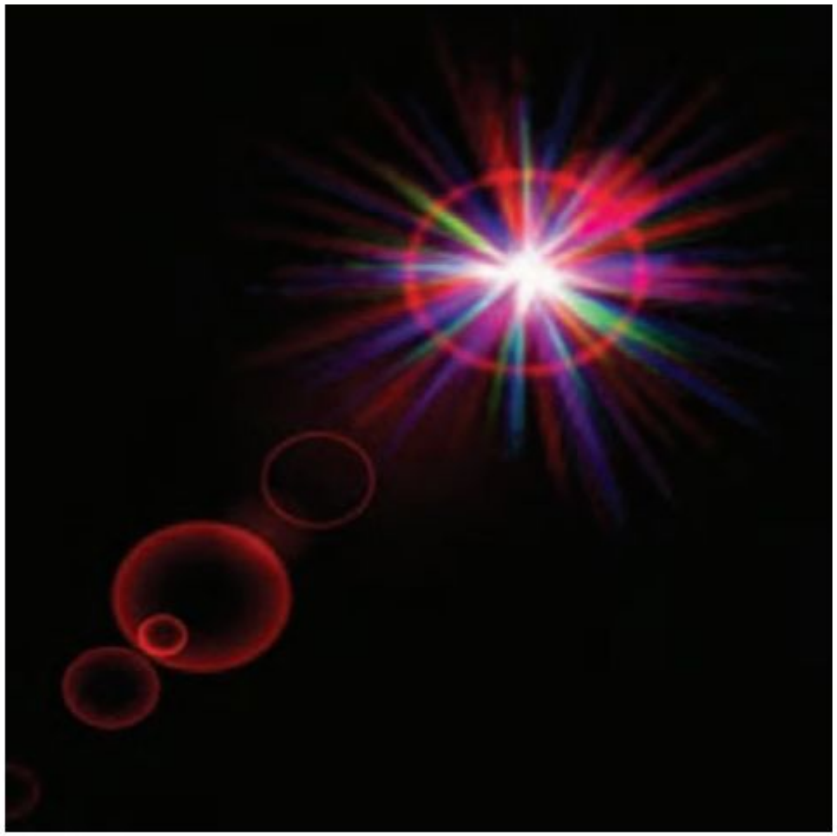
- A estos efectos se los suele llamar “efectos de deslumbramiento”.

- Estos efectos se  utilizan para dar la impresión de un incremento del brillo en la escena o de los objetos.

- Están presente en gran medida  en fotos y películas.

# Lens Flare y Bloom

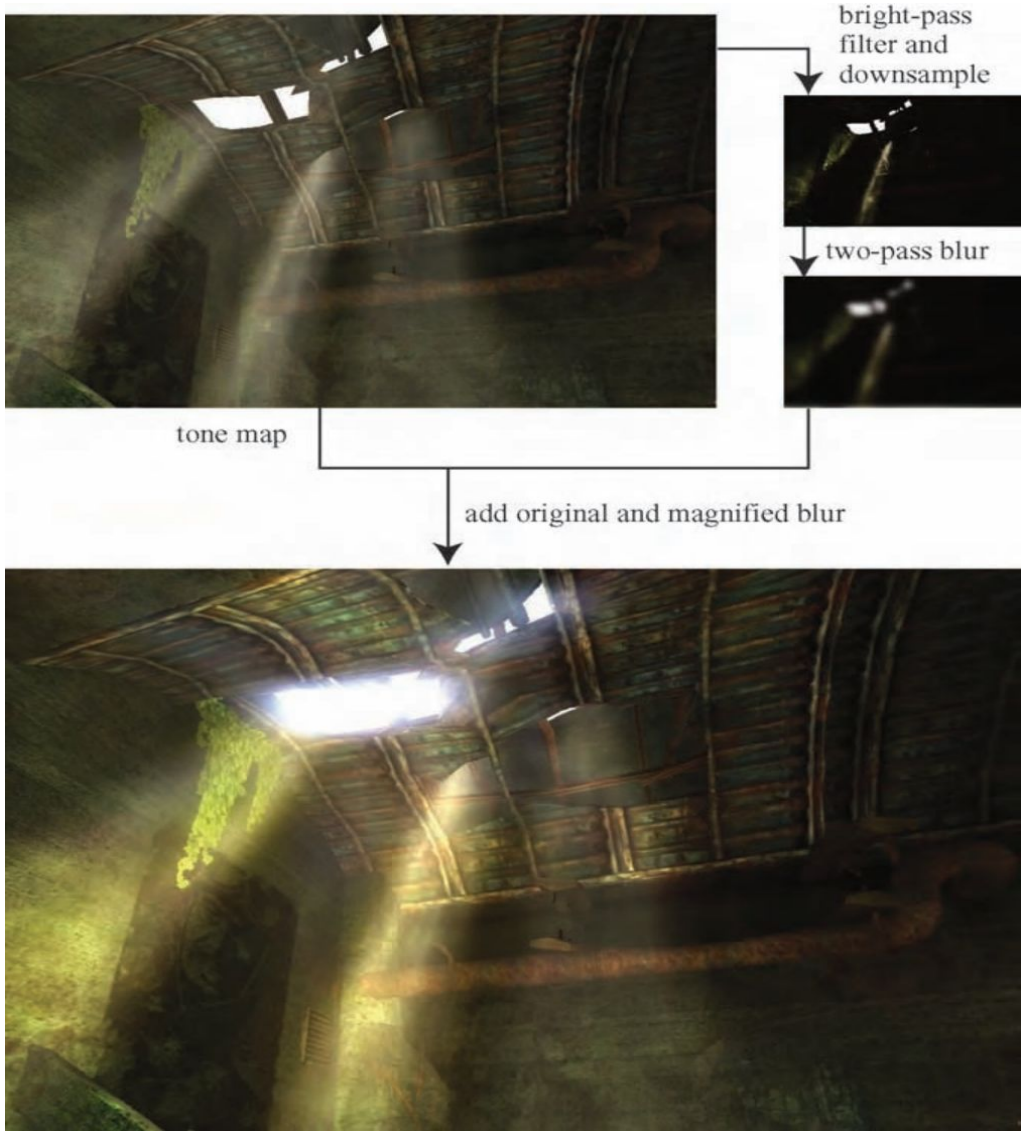
A continuación se pueden ver las texturas que conforman un lens flare. A la derecha, se pueden ver un halo y un bloom en la parte superior y abajo dos texturas brillantes. A estas texturas luego se les da color cuando se renderizan



# Lens Flare y Bloom

Efectos de Lens Flare y bloom, además también se tienen filtros de profundidad de campo y motion blur

# Lens Flare y Bloom



# Lens Flare y Bloom

Como se ve en la imagen superior izquierda, en primer lugar se aplican a la imagen blurs radiales centrados en el sol.
Luego, tal como figura en las dos imágenes de abajo, se le aplican dos pasadas de blurs en serie lo cual deriva en un blur suave y de alta calidad.
Cabe aclarar que estos blurs se realizan a la mitad de resolución para disminuir el costo en tiempo de ejecución del algoritmo.

# Depth of Field

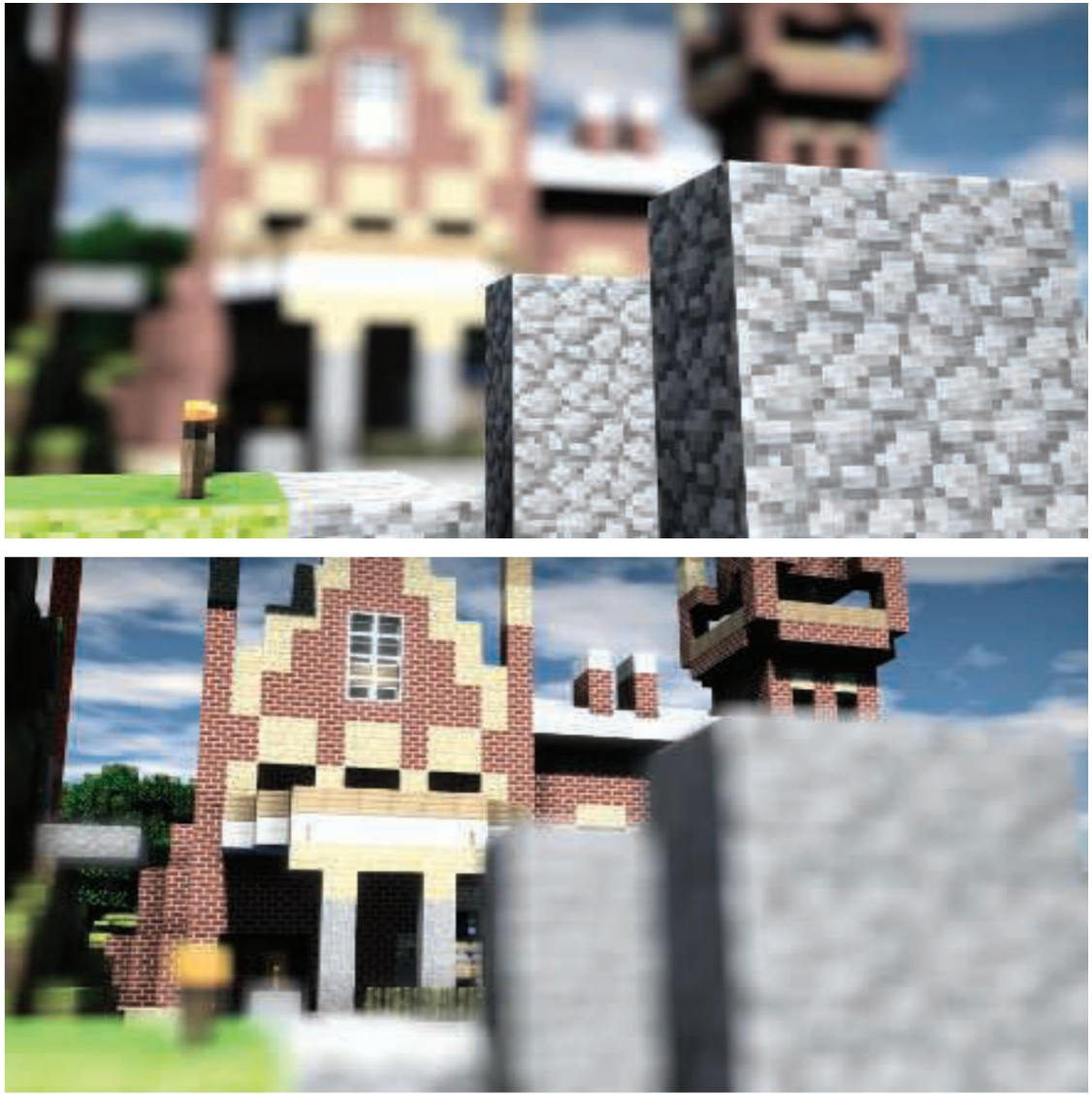
# Depth of Field

**Depth of Field**: Para el lente de una cámara con una configuración dada, existe un rango en el cual los objetos se encuentran “en foco”, a eso le llamamos “depth of field” o por su traducción, “profundidad de campo”.

En la fotografía, este desenfoque viene dado por el tamaño de apertura y el largo del foco.

Reducir el tamaño de la apertura aumenta la profundidad de campo, con lo cual un rango más amplio de profundidades es enfocada, pero a la vez, se disminuye la cantidad de luz que forma la imagen

# Depth of Field

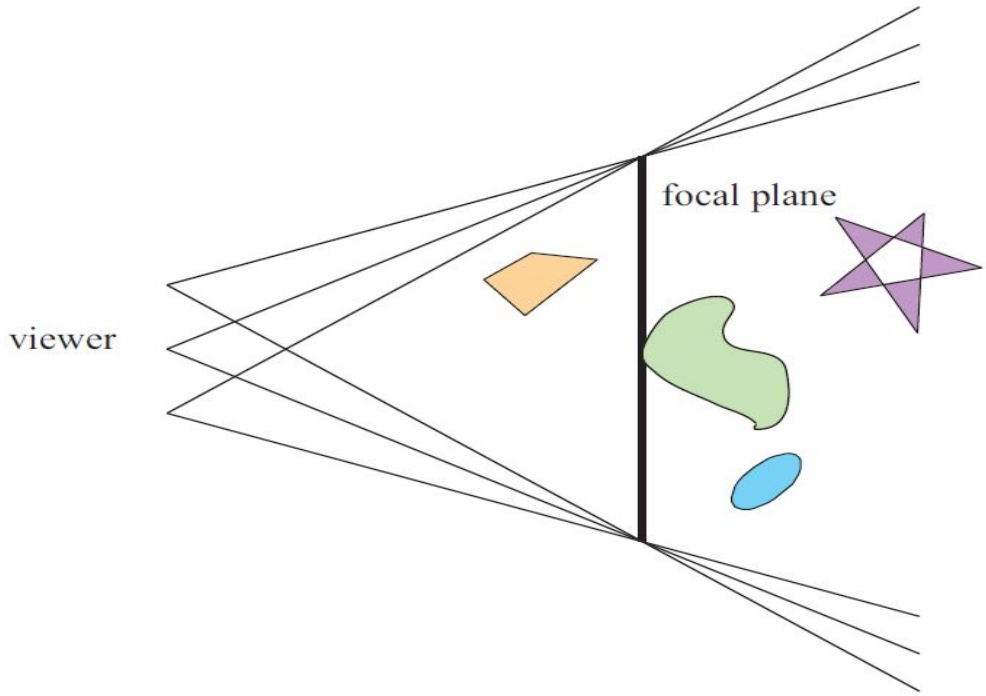


# Depth of Field

Una estrategia que se puede utilizar para simular este efecto de profundidad de campo es utilizar “buffers de acumulación”.

Variando la posición de la vista en la lente y manteniendo el punto de enfoque fijo, los objetos se volverán más borrosos en relación a la distancia que se encuentren del punto focal.

La ubicación del espectador se mueve un poco, manteniendo la dirección de la vista apuntando al punto focal. Cada imagen renderizada se suma y se muestra el promedio de todas las imágenes.




Sin embargo, al igual que con otros efectos de acumulación, este método tiene un alto costo de múltiples representaciones por imagen.

# Depth of Field

Las superficies se pueden clasificar en 3 zonas:

-Las que están en foco cerca de la distancia del plano focal (*focus field* o *mid-field*)

-Las que están por detrás del plano focal (*far-field*)

-Las que están más cerca que el plano focal (*near-field*)

Para una superficie en la zona del focus field se tiene que dicha superficie está en foco, ya que todas las imágenes acumuladas tienen aproximadamente el mismo resultado. Se dice que puede tener un desenfoque de menos de medio píxel.

Debido a esto es que el depth of field se refiere a realizarle un desenfoque a las zonas del far-field y el near-field

# Depth of Field

Una solución encontrada para representar el depth of field es crear capas separadas de la imagen. Es decir, renderizar una imagen que contenga solamente los objetos que se encuentran en foco, una que contenga los que se encuentran en el far-field y otra que contenga los del near-field.
Finalmente las 3 imágenes se componen juntas desde atrás hacia adelante.

A este método se le suele llamar “enfoque de 2.5 dimensiones” debido a que a imágenes bidimensionales se les dan profundidades y luego son combinadas para dar una sensación realista de profundidad.

Una desventaja de este método es que se podrían generar muchas imágenes si existen objetos en la escena que cambien de estar en foco a estar desenfocado abruptamente.

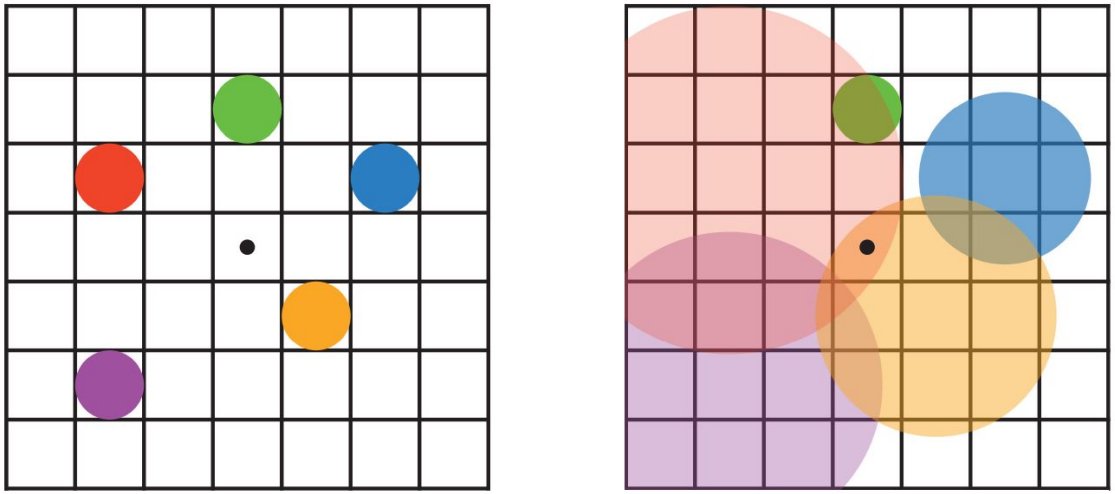
Además, otro problema que tiene es que los objetos tienen un blur uniforme independientemente de variaciones en la distancia al plano focal

# Depth of Field

# Depth of Field

Depth of field aplicado en videojuegos, donde se ve que el far-field y el near-field se mezclan suavemente con el focus field.

# Depth of Field



En la imagen de arriba se muestran los círculos de confusión superpuestos.
A la izquierda hay una escena con cinco puntos, todos enfocados.

Imaginemos que el punto rojo está más cerca del espectador, en el near-field, seguido del punto naranja;
el punto verde está en el focus field; y los puntos azul y violeta están en el far field, en ese orden.

La figura de la derecha muestra los círculos de confusión que resultan de aplicar la profundidad de campo, donde un círculo más grande tiene un menor efecto por píxel.

El verde no ha cambiado, ya que está enfocado.

El píxel central se superpone solamente por los círculos rojo y naranja, por lo que estos se mezclan, rojo sobre naranja, para darle el color al píxel.

# Depth of Field

Aquí se puede ver un ejemplo de near-field blur.

A la izquierda está la imagen original sin efecto de profundidad de campo.

En el medio, los píxeles en el near-field están borrosos, pero tienen un borde nítido donde están adyacentes al focus field. Es decir, las aristas adyacentes a zonas dentro del foco no se ven borrosas sino nítidas.

La derecha muestra el efecto de usar una imagen de near-field separada compuesta por encima del contenido más distante

# Depth of Field

En la imagen de arriba se ve la profundidad de near y far field con círculo de confusión pentagonal en el poste reflectante brillante en el primer plano.

# Motion Blur

# Motion Blur

En una película, el “motion blur” o “desenfoque de movimiento” es generado por el movimiento de un objeto por la pantalla durante el transcurso de un frame o también es generado por el movimiento de la cámara.

Esta técnica es bien conocida por representar alto grado de realismo los movimientos de los objetos de la escena así como también de los movimientos de la cámara.

Los objetos que se mueven rápidamente parecen espasmódicos sin desenfoque de movimiento, "saltando" por muchos píxeles entre fotogramas. Esto se puede considerar como un tipo de aliasing, pero de naturaleza temporal más que espacial.

El desenfoque de movimiento se puede considerar como antialiasing en el dominio del tiempo.

# Motion Blur

El desenfoque de movimiento depende del movimiento relativo. Si un objeto se mueve de izquierda a derecha a lo largo de la pantalla, aparece borroso horizontalmente en la pantalla.
Si la cámara está rastreando un objeto en movimiento, el objeto no se difumina, sino que el fondo lo hace.

La cámara está fija y el auto está borroso.

La cámara sigue al auto y es el fondo el que está borroso.

# Motion Blur

-De forma similar a depth of field, la acumulación de una serie de imágenes proporciona una forma de crear desenfoque de movimiento.

-Durante un frame, la escena es renderizada varias veces, con la cámara y los objetos reposicionados para cada vez. Las imágenes resultantes se mezclan, dando una imagen borrosa donde los objetos se mueven en relación al punto de vista de la cámara.

-Para la renderización en tiempo real, este proceso es normalmente contraproducente, ya que puede reducir considerablemente los FPS.

-Existen diferentes fuentes de desenfoque de movimiento, estos se pueden clasificar como:

- cambios de orientación de la cámara
- cambios de posición de la cámara
- cambios de posición de un objeto
- cambios de orientación de un objeto

# Motion Blur

Para poder utilizarlo de forma eficiente, la idea es transformar la ubicación y profundidad en la pantalla de un píxel a una ubicación espacial mundial, luego transformar este punto mundial usando la cámara del frame anterior a una ubicación de pantalla.

La diferencia entre estas ubicaciones del espacio de pantalla es el vector de velocidad, que se utiliza para desenfocar la imagen para ese píxel.

# Motion Blur

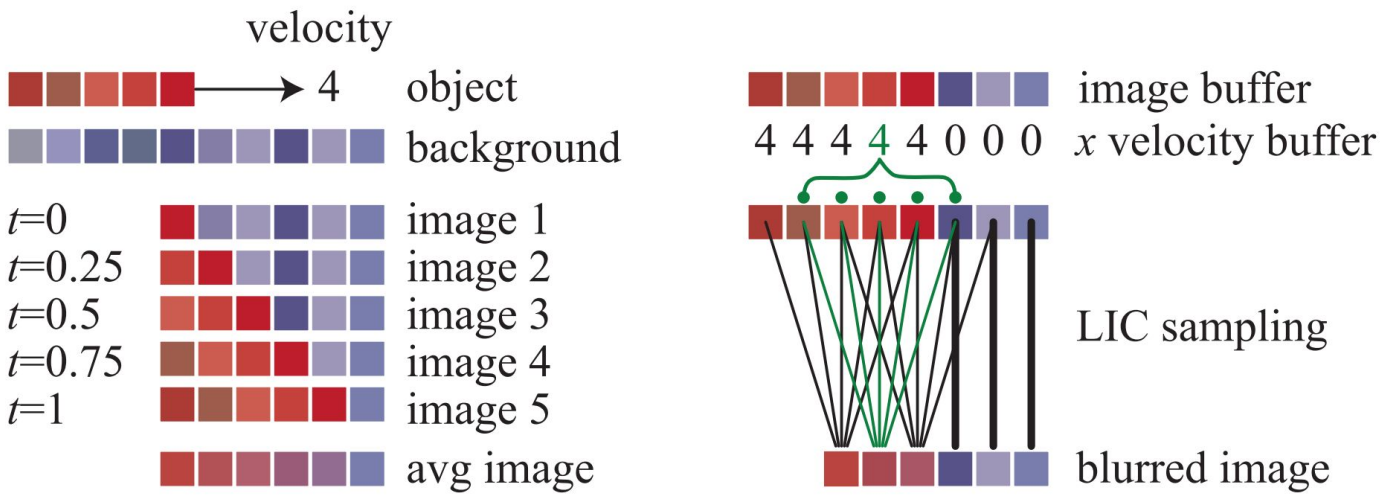
Blur radial centrado en el personaje

# Motion Blur

# Motion Blur

Una forma para poder aplicar el motion blur es conociendo la velocidad de la superficie de cada píxel. Esta información se puede obtener mediante la utilización de un "*buffer de velocidad*"

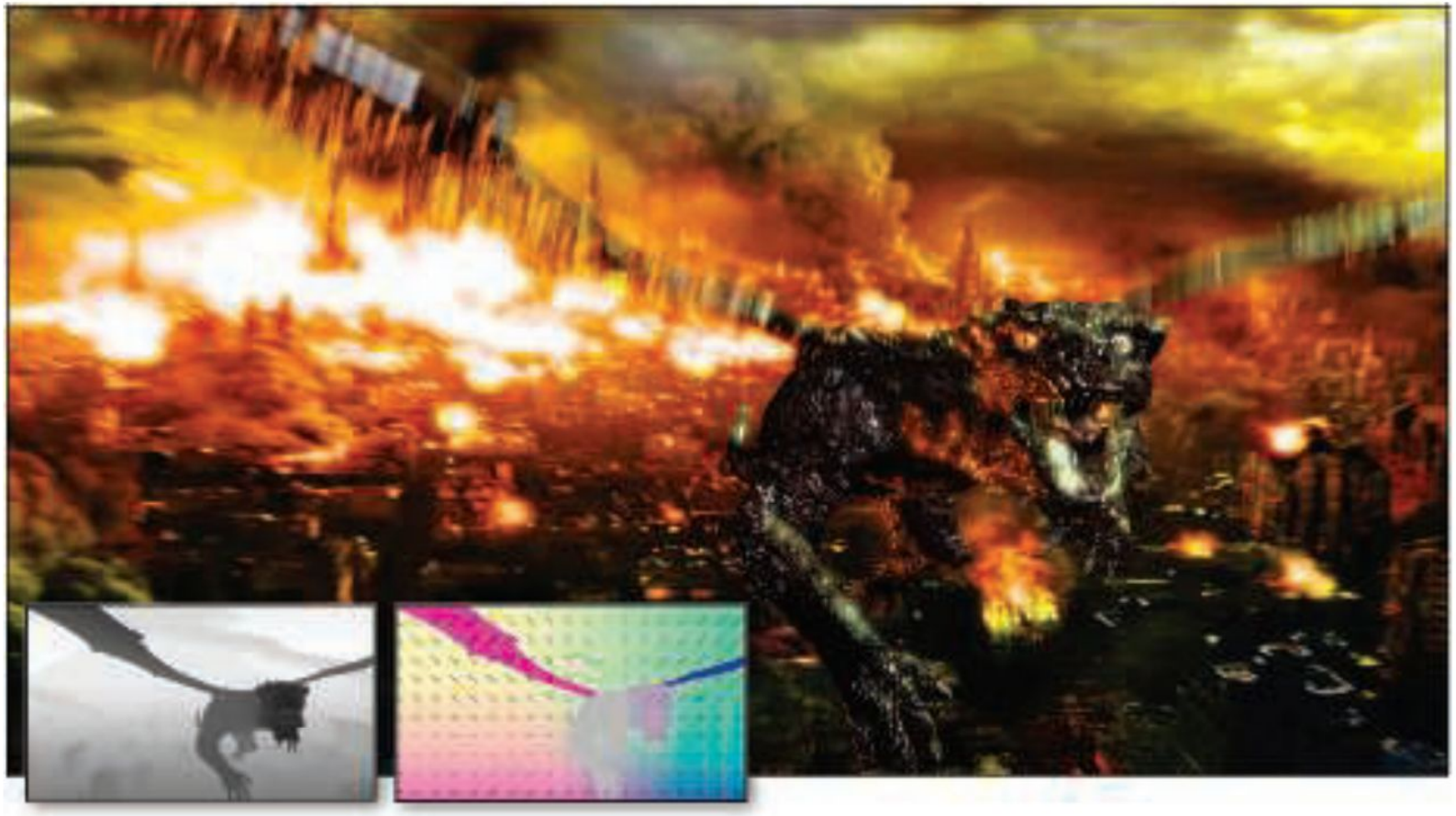
A continuación se presenta un ejemplo de la utilización de un buffer de velocidad en comparación a un buffer de acumulación




A la izquierda se visualiza un renderizado de buffer de acumulación. El conjunto rojo de píxeles representa un objeto que se mueve cuatro píxeles hacia la derecha en un solo fotograma. Los resultados de seis píxeles en los cinco fotogramas se promedian para obtener el resultado final correcto en el fondo.

A la derecha, una imagen y un buffer de velocidad en la dirección *x* que se generan en el momento 0.5 (los valores del búfer de velocidad en *y* son todos ceros, ya que no hay movimiento vertical).
El búfer de velocidad se utiliza para determinar cómo se muestrea el buffer de imagen. Cinco muestras, una por píxel, son tomadas y promediadas.

# Motion Blur



La imagen de arriba muestra el motion blur a causa de movimientos de objeto y de cámara. Además, se muestran los buffer de profundidad y velocidad en la parte inferior izquierda.

# Motion Blur

FIN