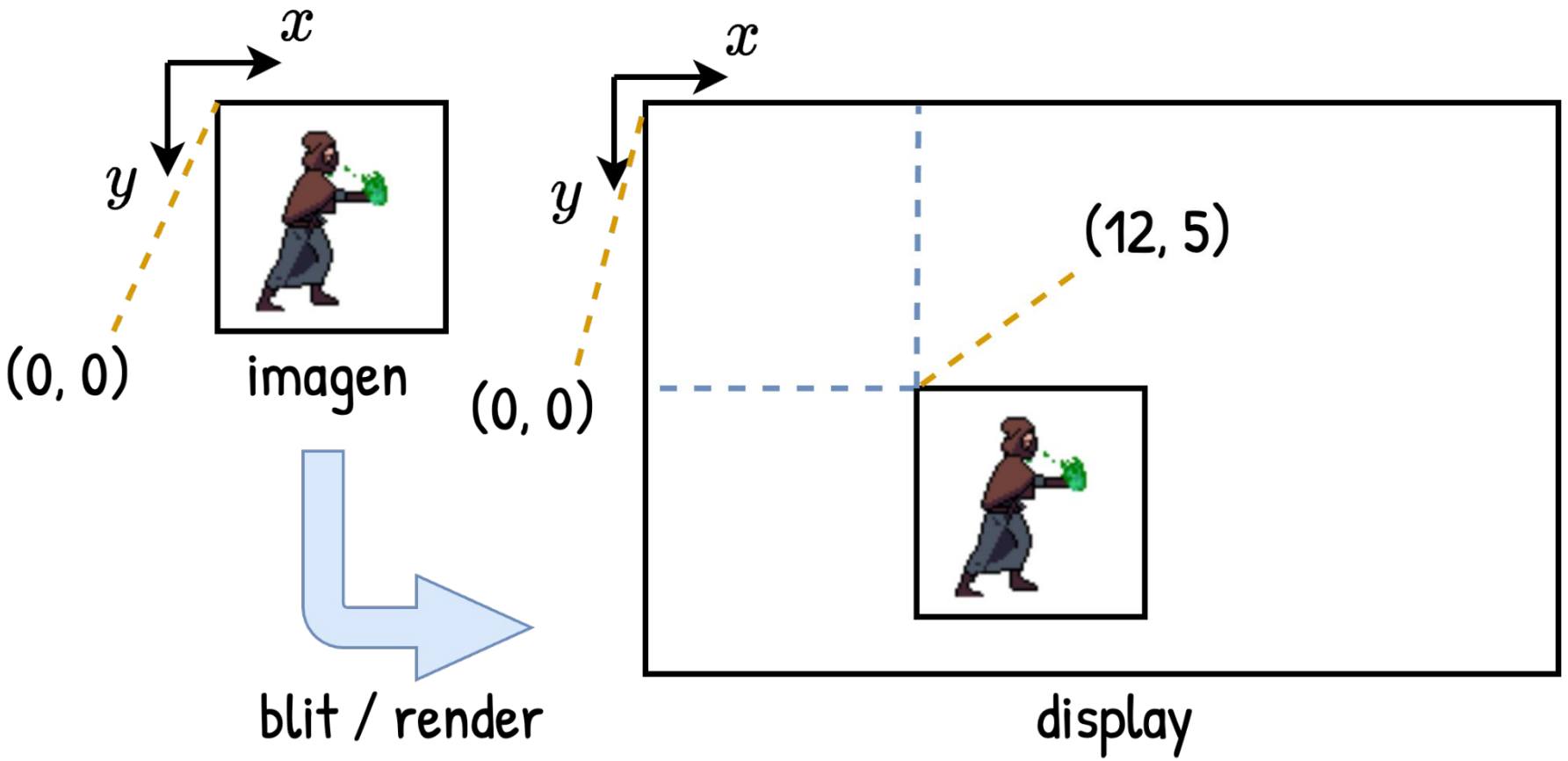
Interfaz gráfica (intro)

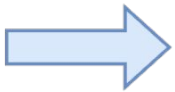
Martin Di Paola

Facultad de Ingeniería
Universidad de Buenos Aires

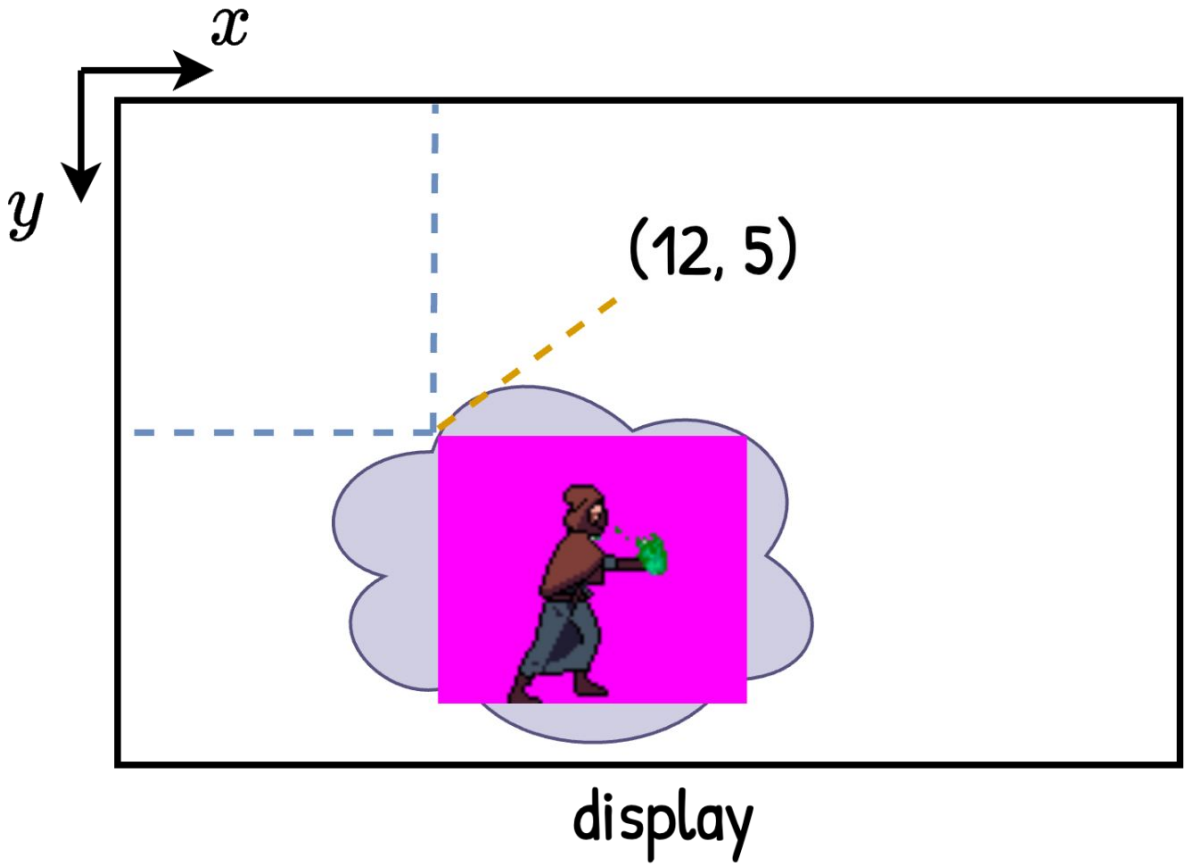
Imagenes (src & dst)



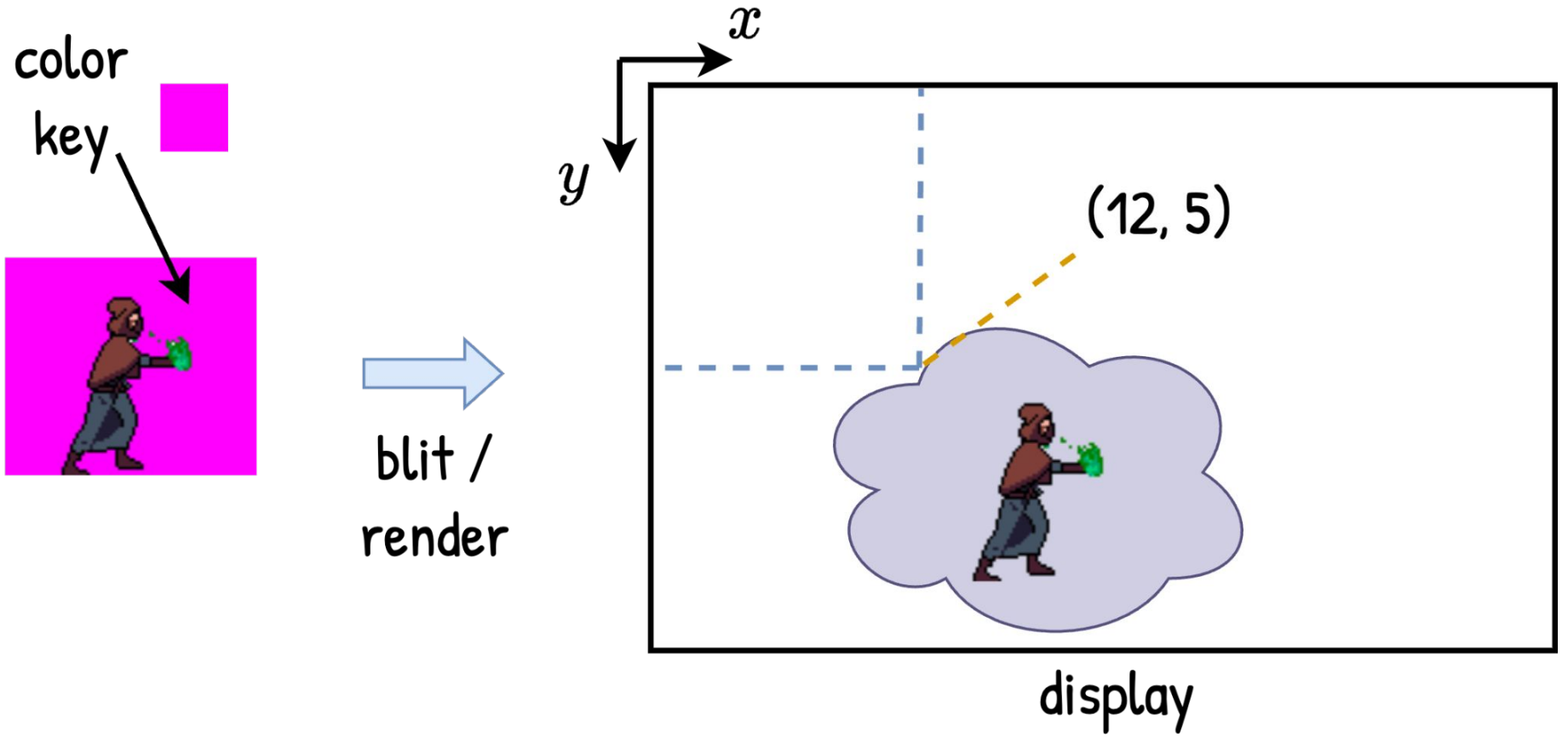
Las imagenes son matrices



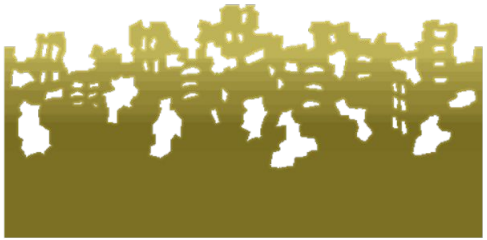
blit /
render



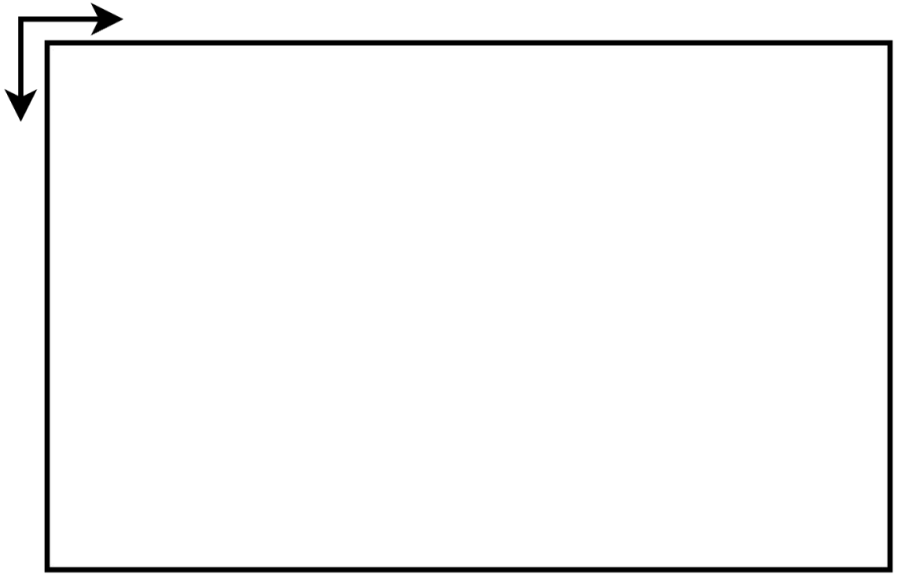
Color key



Composición de la escena (renderizado del frame)

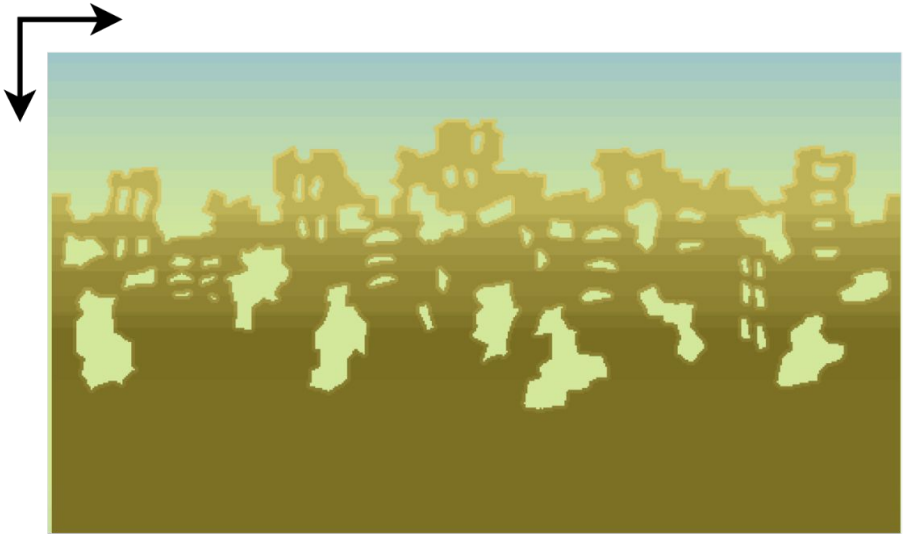


????

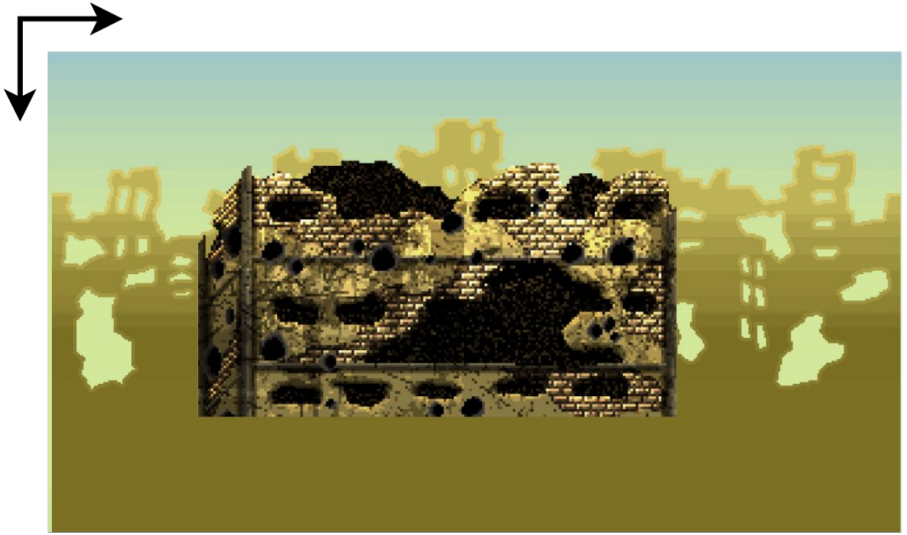


Algoritmo del pintor (aka: renderizar del fondo hacia adelante)

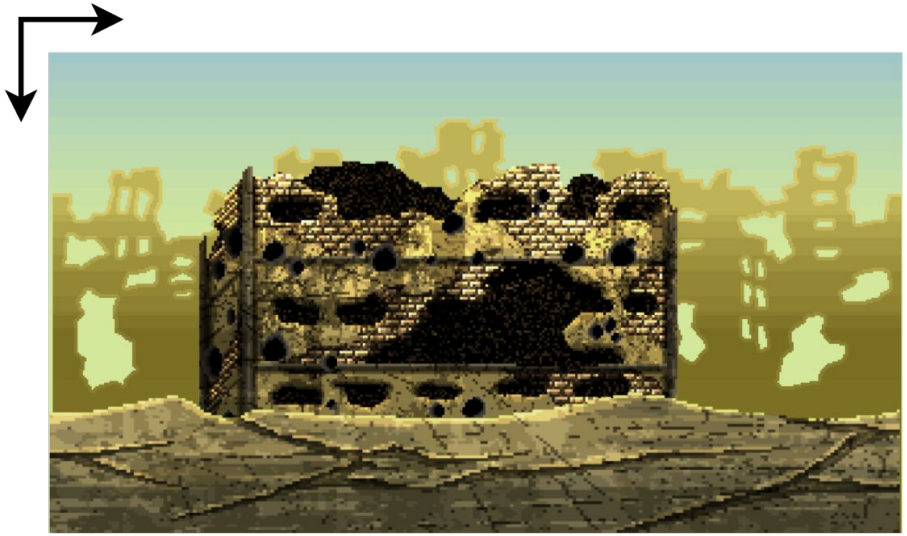
Algoritmo del pintor (aka: renderizar del fondo hacia adelante)



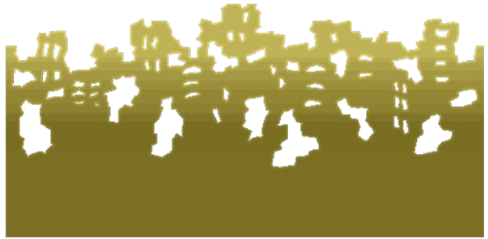
Algoritmo del pintor (aka: renderizar del fondo hacia adelante)

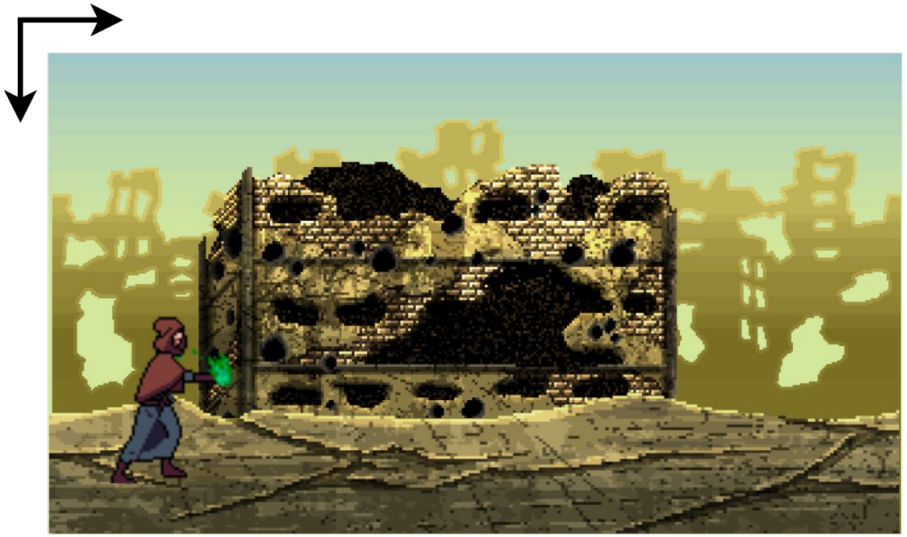


Algoritmo del pintor (aka: renderizar del fondo hacia adelante)

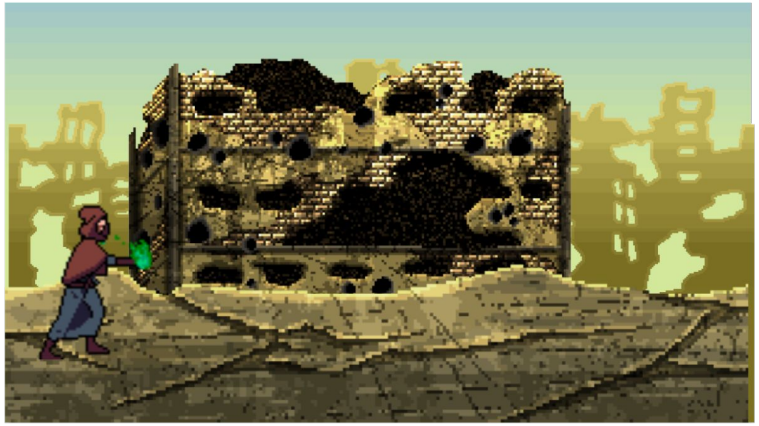


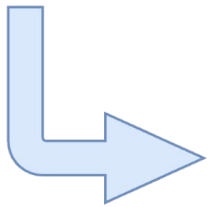
Algoritmo del pintor (aka: renderizar del fondo hacia adelante)



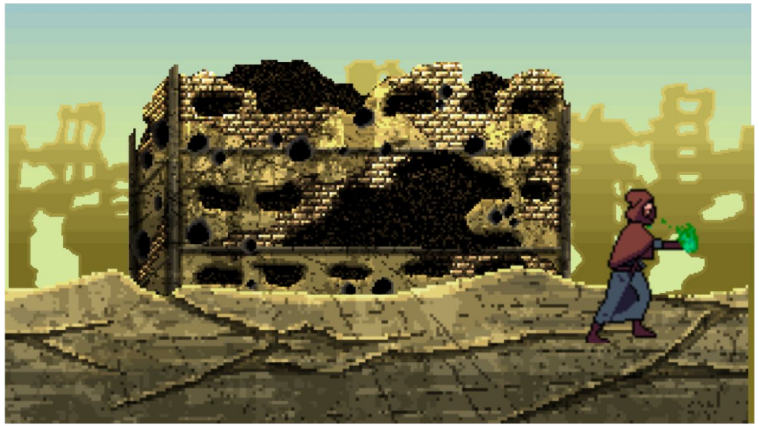


Movimiento de una imagen / actualización del escena

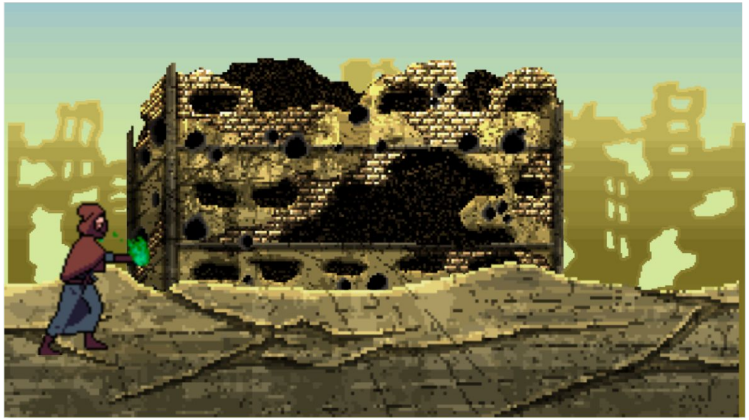


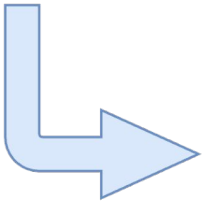


????

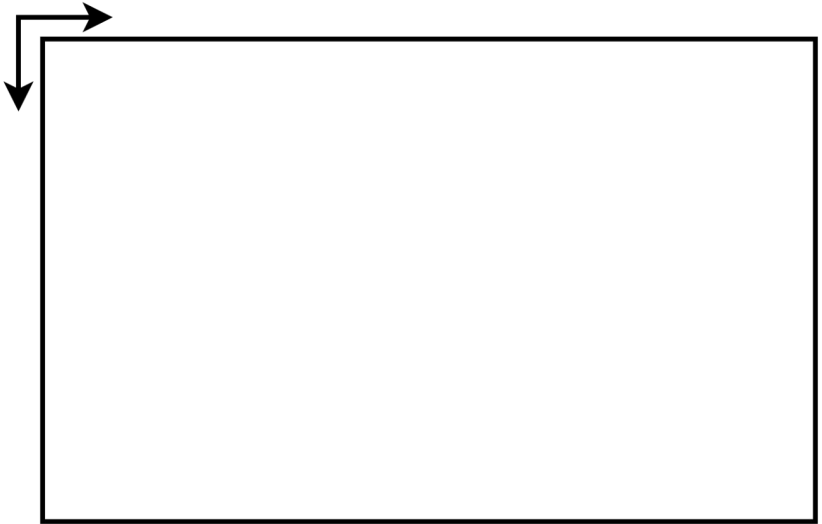


Clear





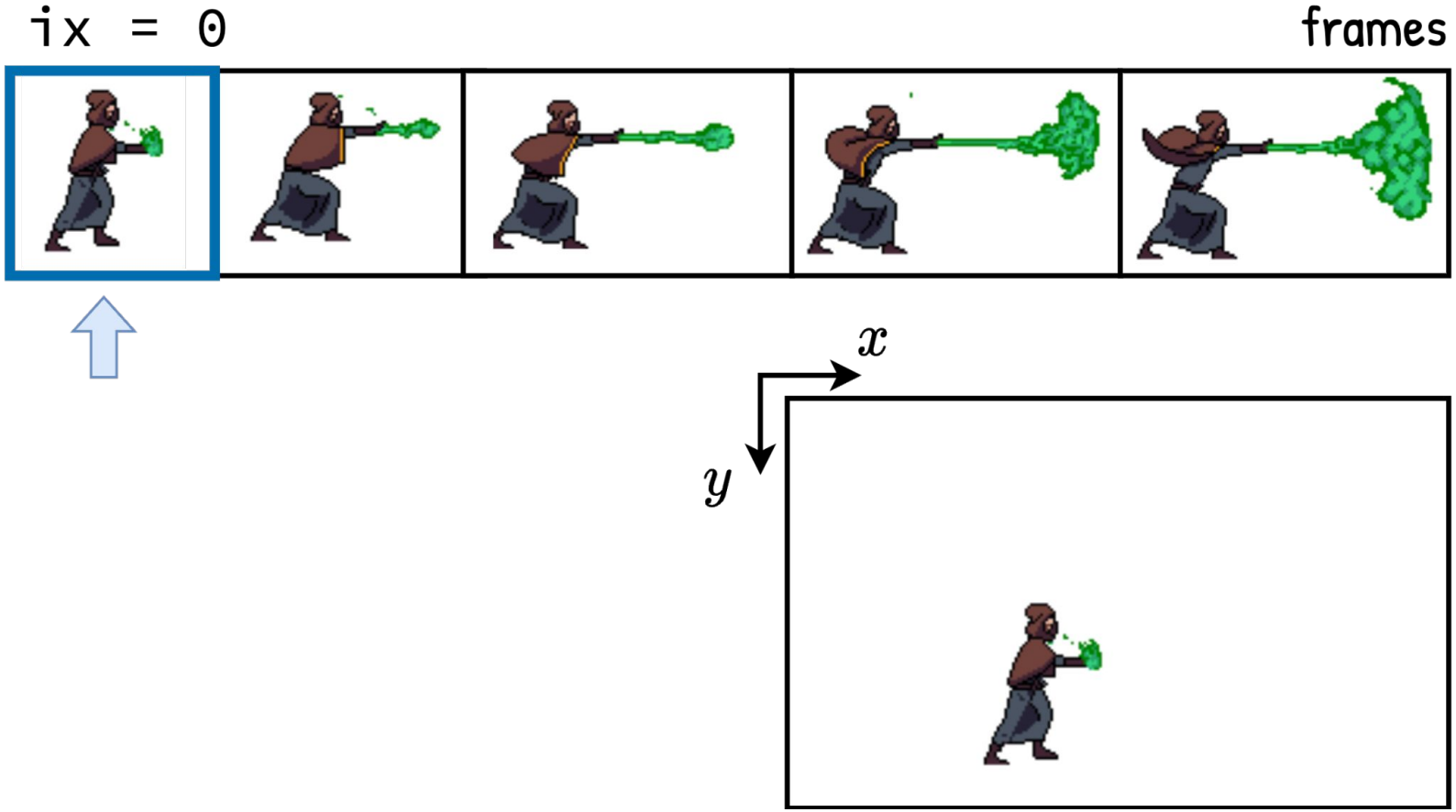
clear / fill



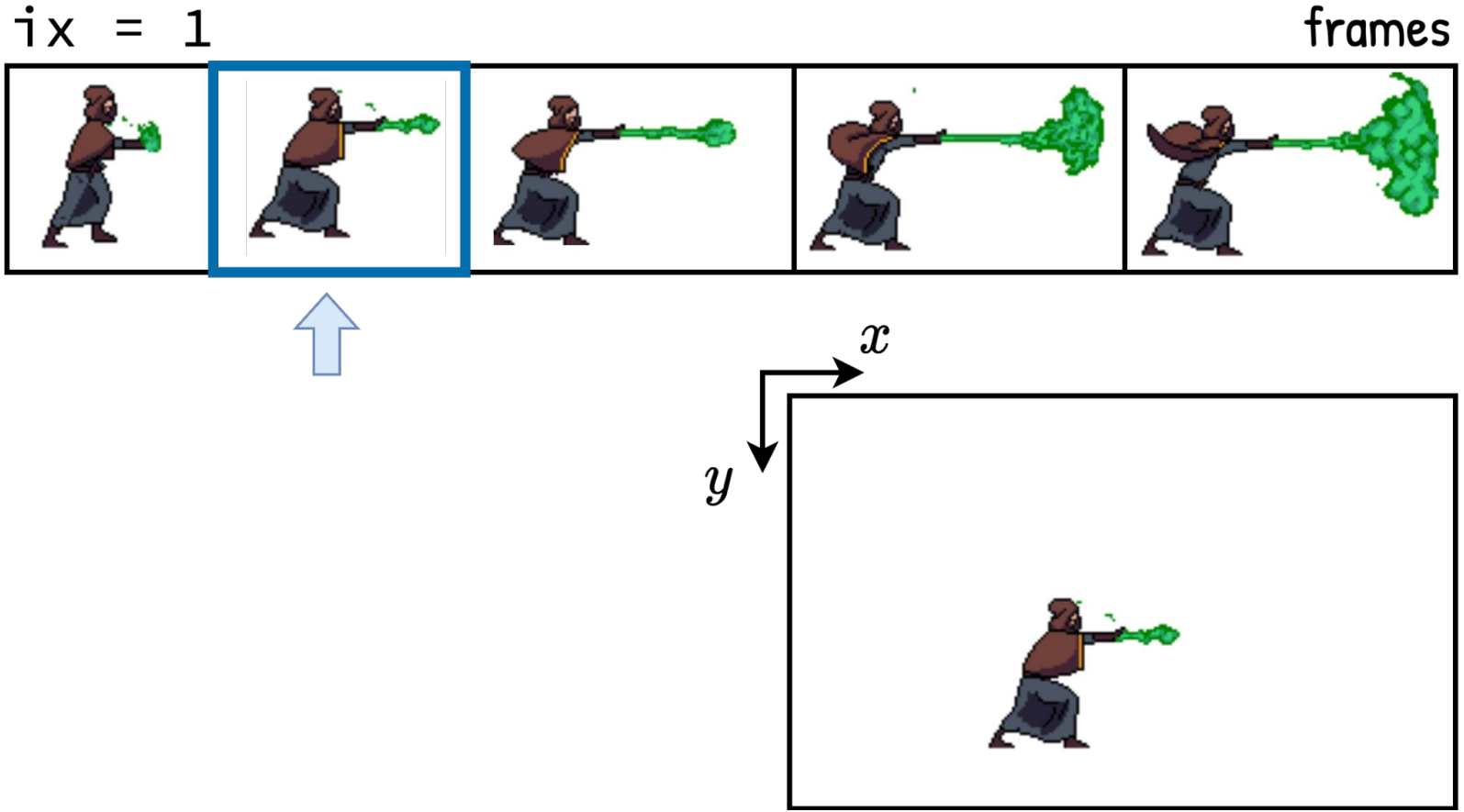
Main loop: render

```
void main_loop() {
  //init_resources();
  while (not exit) {
    //update_logic_and_physics();
    clear_display();
    //update_animation_frames(it);
    render_in_z_order();
    //it = sleep_and_calc_next_it();
  }
}
```

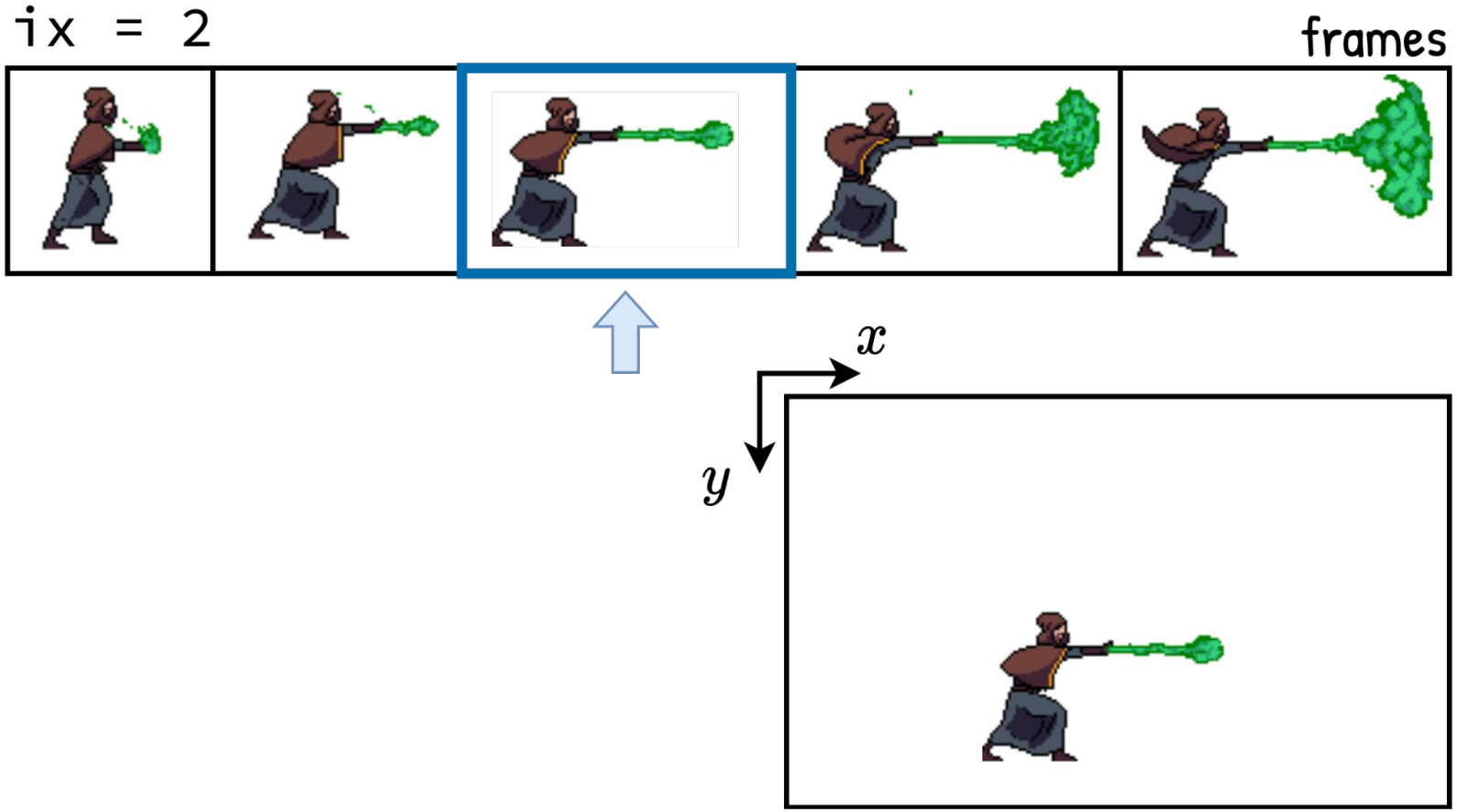
Animaciones



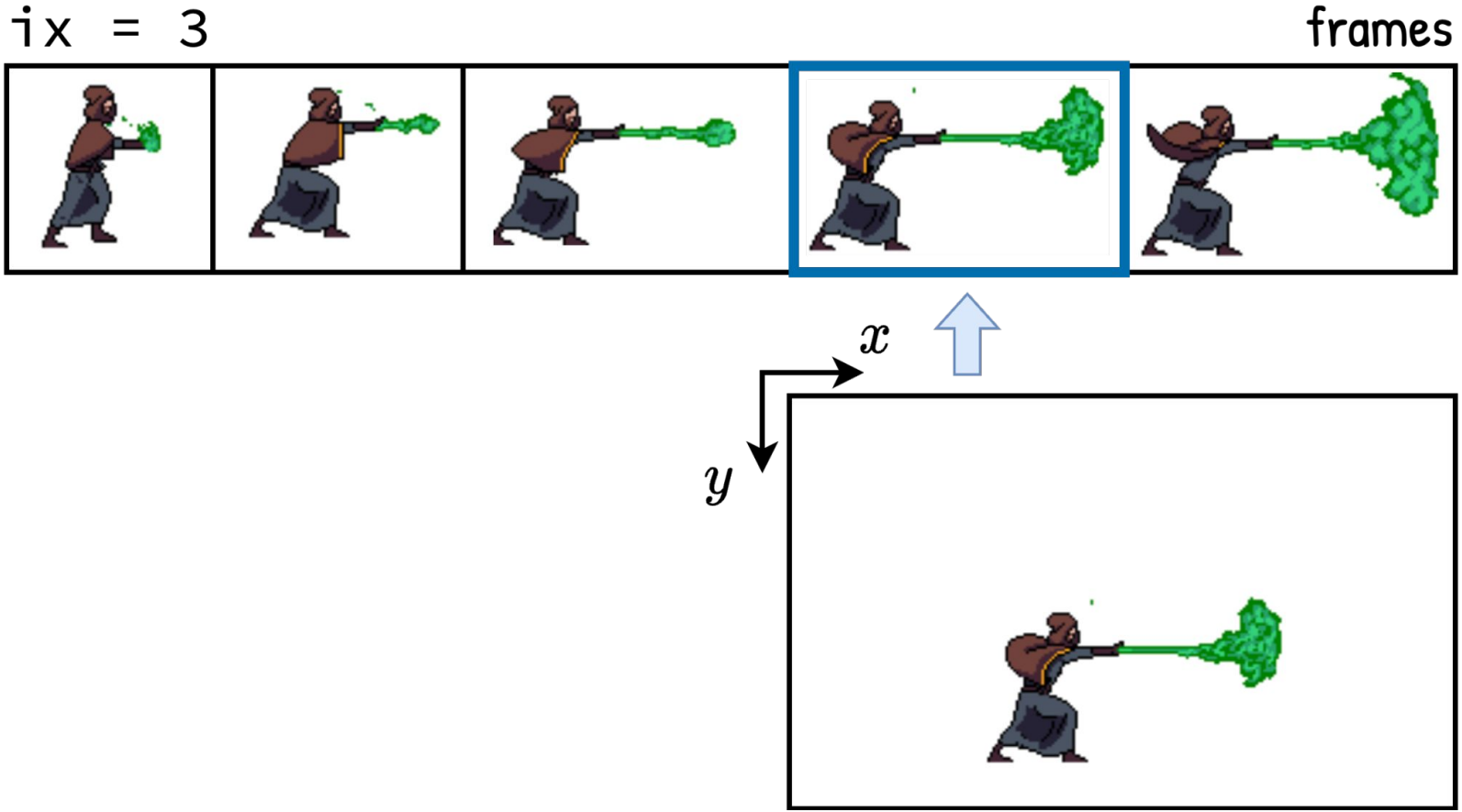
Animaciones



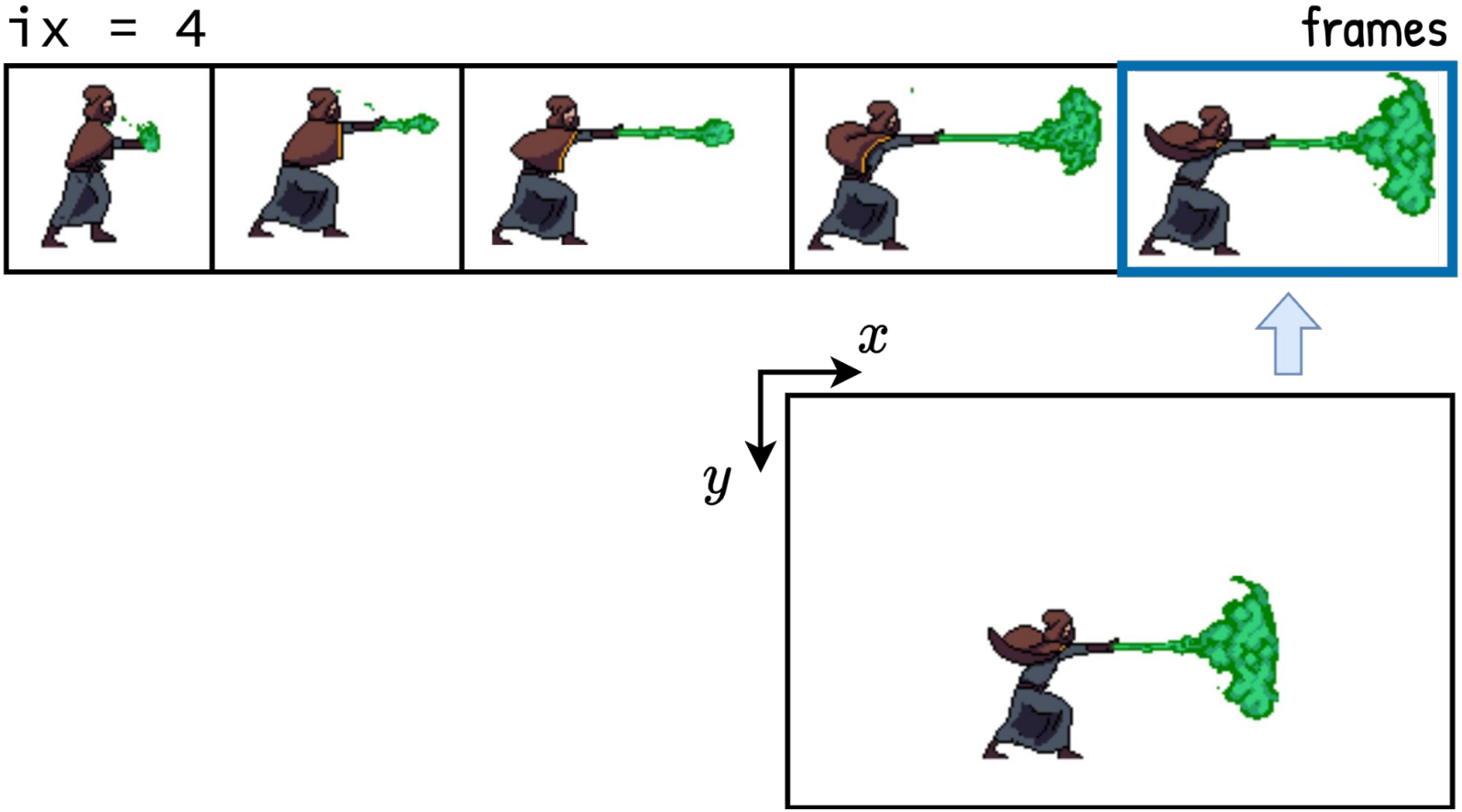
Animaciones



Animaciones



Animaciones



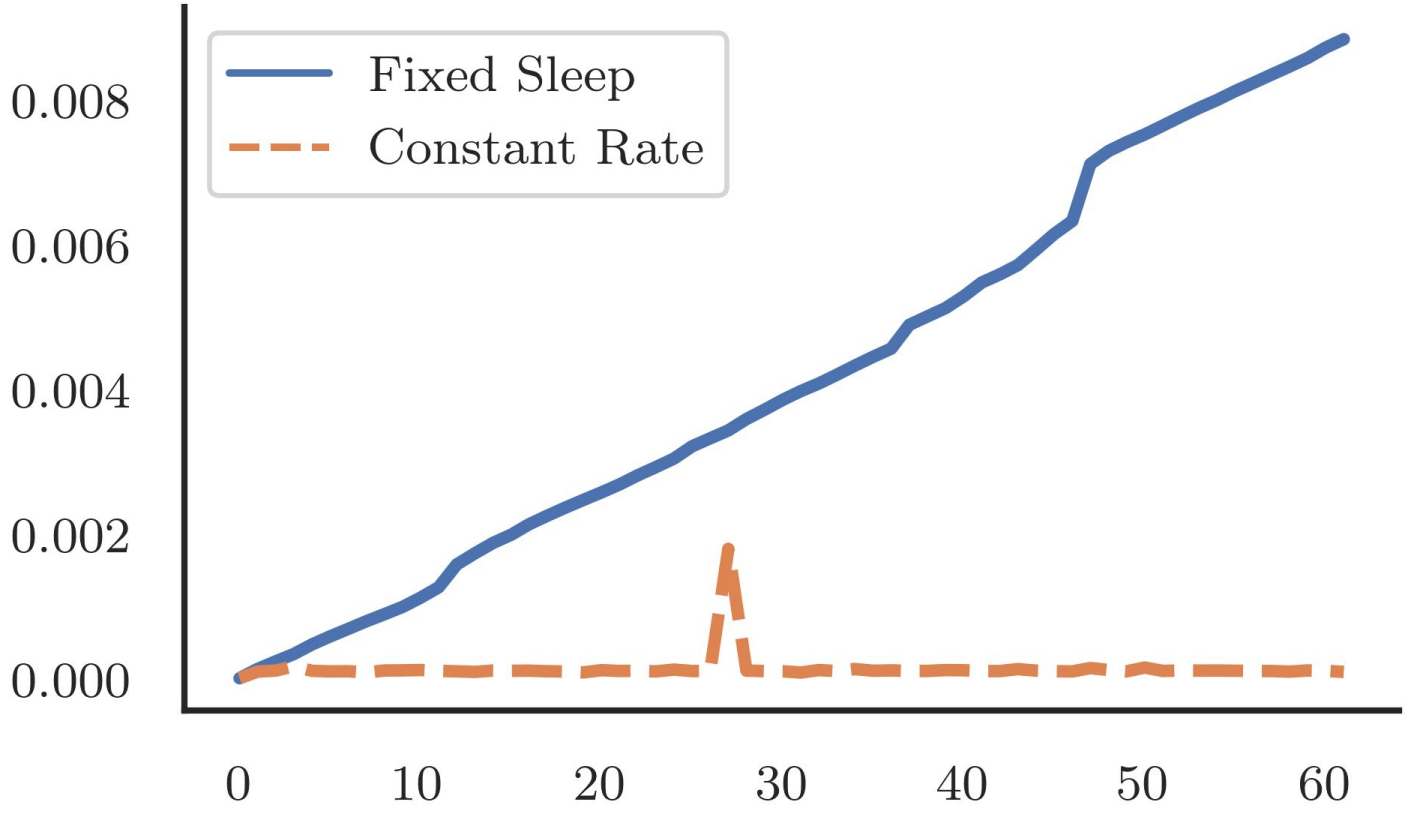
Main loop: animaciones

```
void main_loop() {
  //init_resources();
  while (not exit) {
    //update_logic_and_physics();
    clear_display();
    update_animation_frames(it);
    render_in_z_order();
    ++it;
  }
}
```

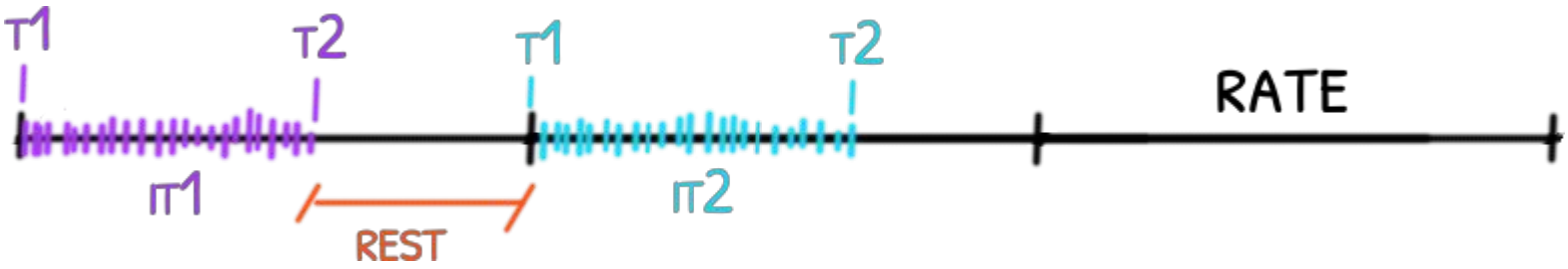
Main loop: sleep -> frames per second (incorrecto)

```
void main_loop() {
  //init_resources();
  while (not exit) {
    //update_logic_and_physics();
    clear_display();
    update_animation_frames(it);
    render_in_z_order();
    ++it;  sleep(FPS);
  }
}
```

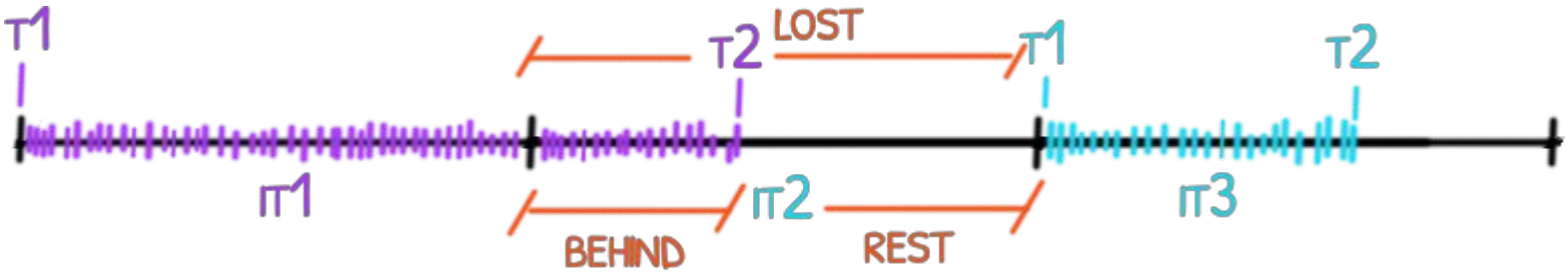
Clock/animacion drift (desfazaje/atraso)



Rest/sleep entre renderizado de frames (iteraciones del render loop)



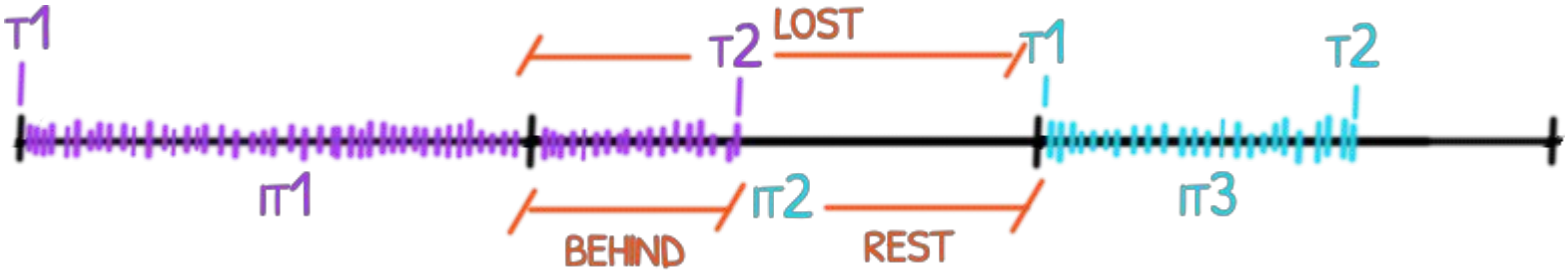
Y si el renderizado tarda más de lo esperado?

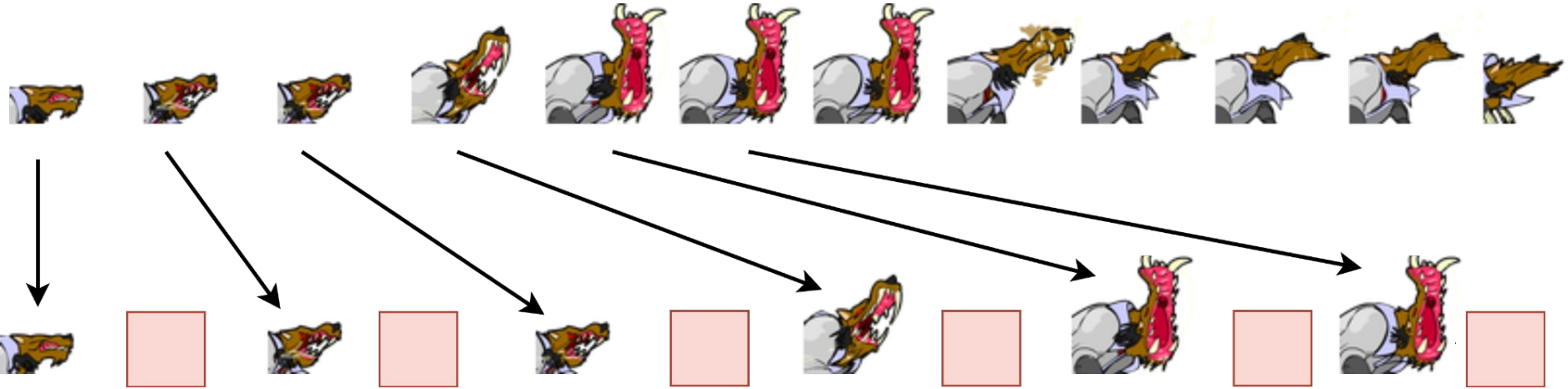


Main loop (I)

```
void main_loop() {
  //init_resources();
  while (not exit) {
    //update_logic_and_physics();
    clear_display();
    update_animation_frames(it);
    render_in_z_order();
    ++it;  if (not behind) { sleep(FPS-x) };
  }
}
```

Drop de frames (incorrecto)

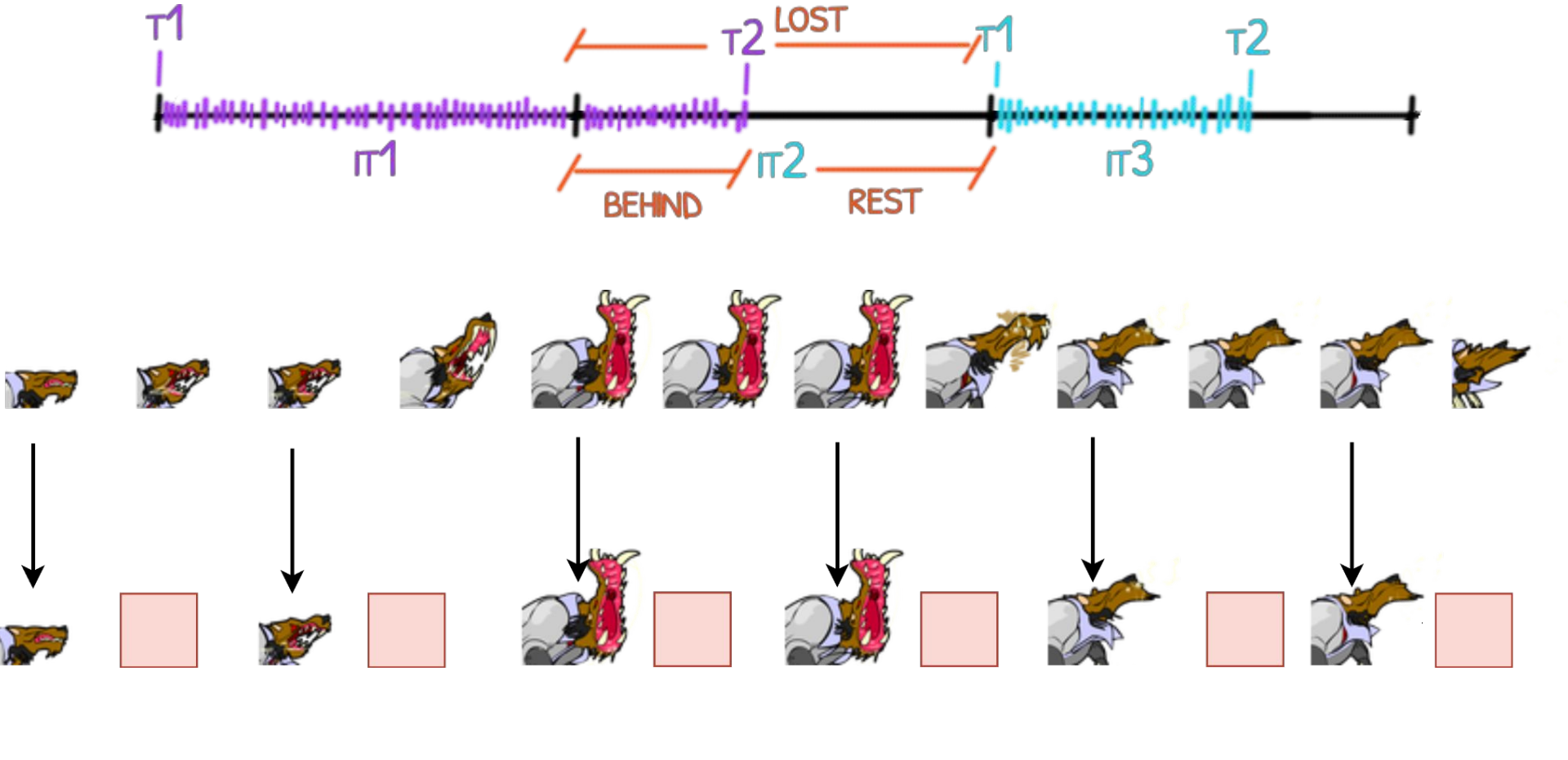




Main loop (I)

```
void main_loop() {
  //init_resources();
  while (not exit) {
    //update_logic_and_physics();
    clear_display();
    update_animation_frames(it);
    render_in_z_order();
    it = sleep_and_calc_next_it(FPS, x);
  }
}
```

Drop de frames (correcto)

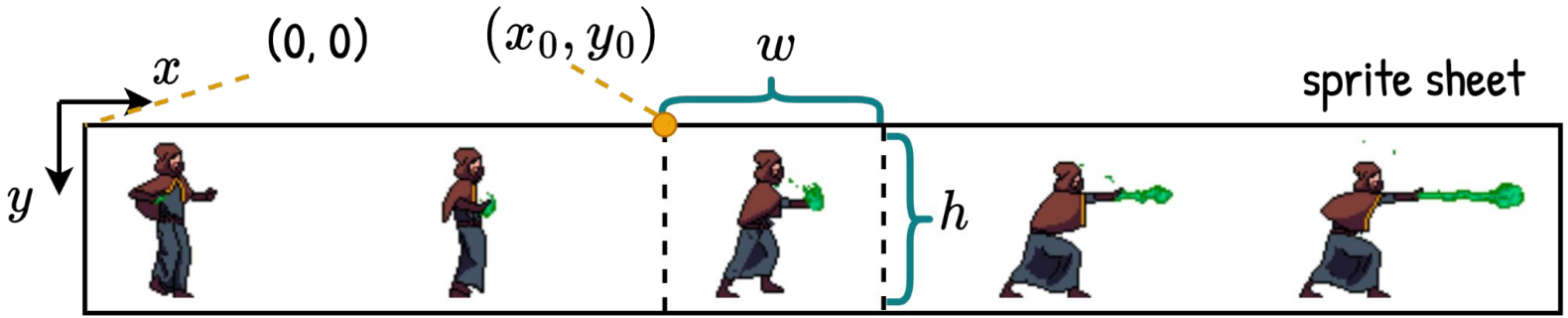


Velocidad de animación en función de la iteración, no del tiempo

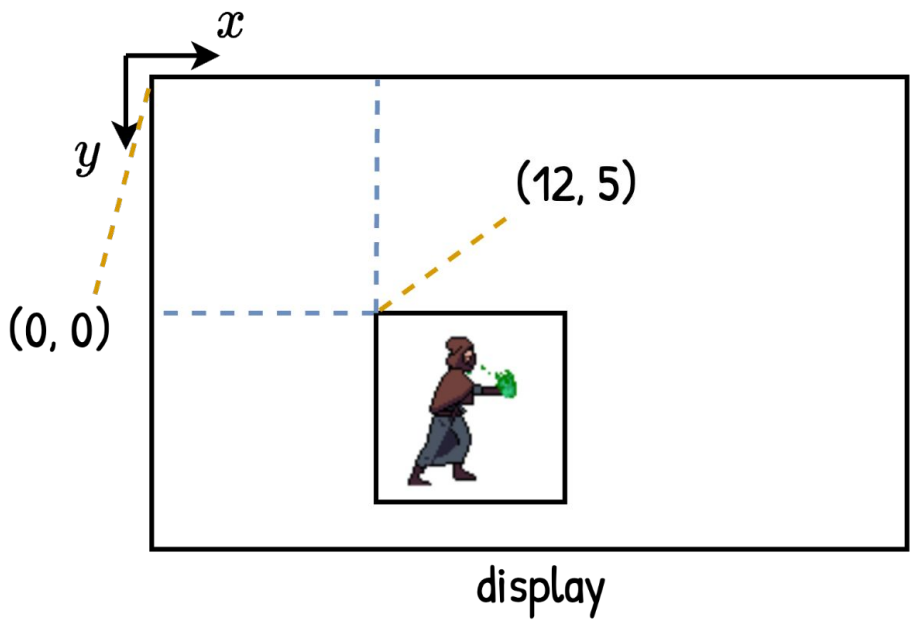
```
def render_foo(it) {
    imgs[it % N]
}

def render_bar(it) {
     imgs[it/2 % N]
}
```

Sprite sheet + clip

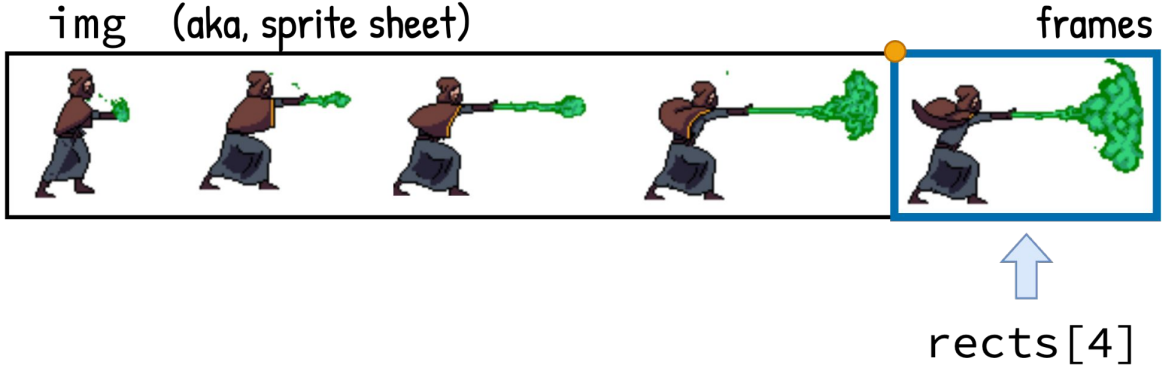


clip
+
blit / render

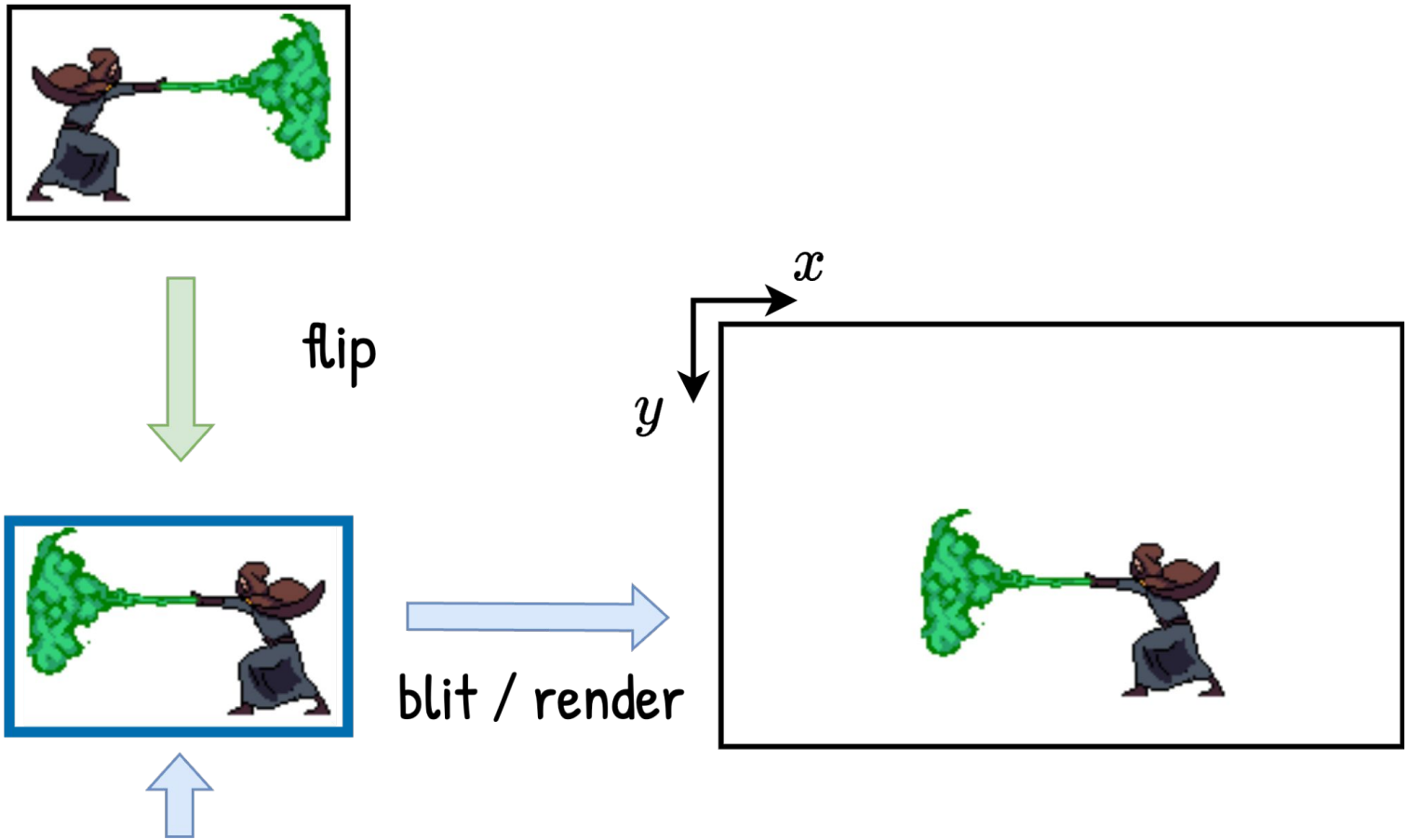


Frames de animación (2 implementaciones)

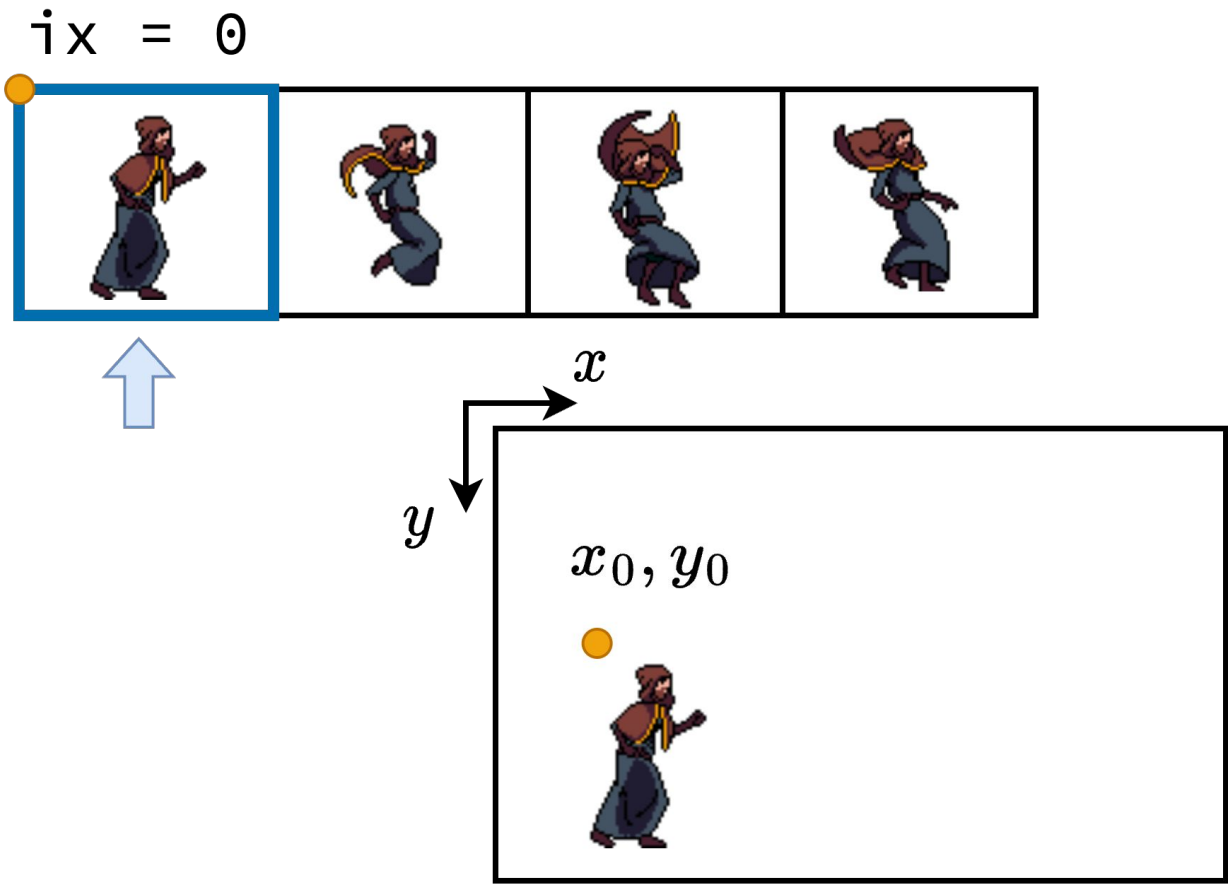
rects[] = $\{(x_0, y_0, h_0, w_0),\ (x_1, y_1, h_1, w_1),\ \ldots\}$



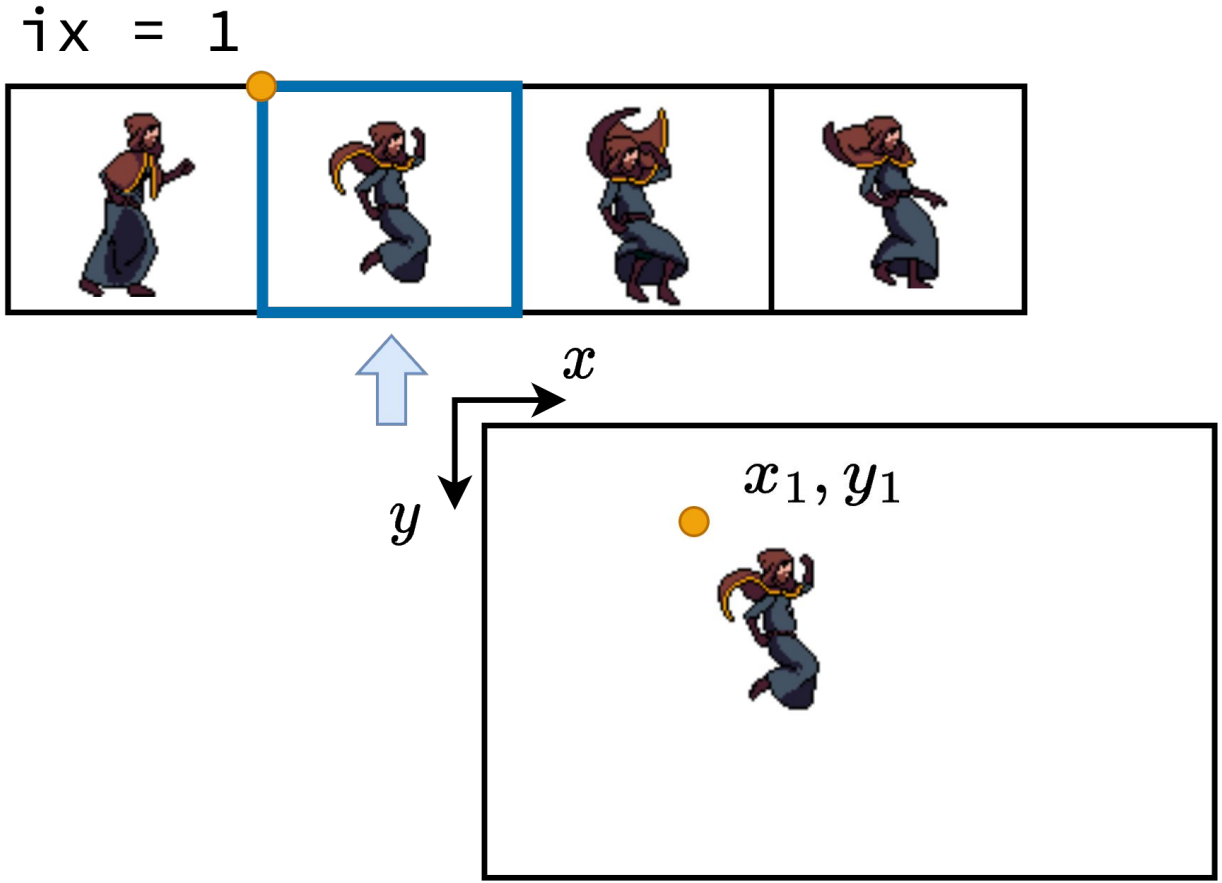
Flip



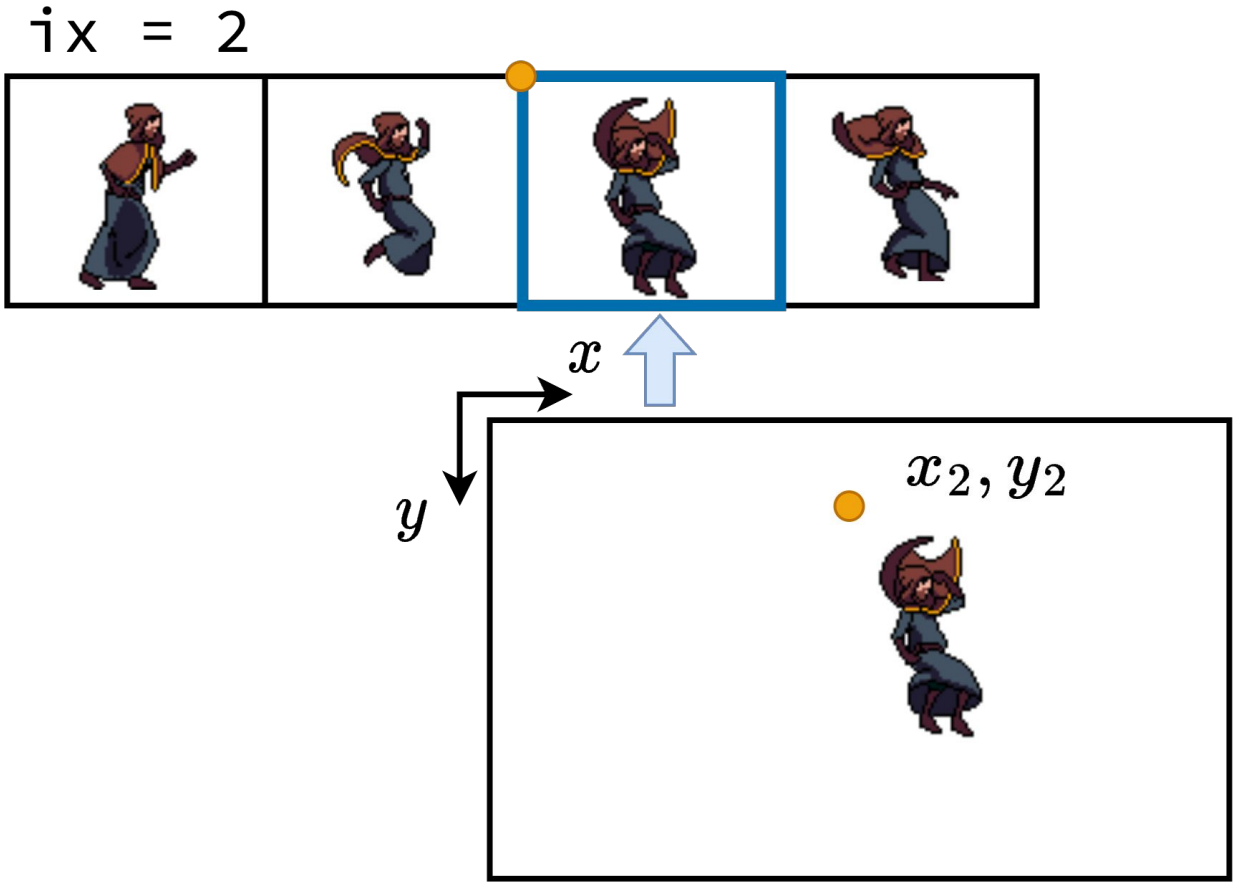
Movimiento: cambio de frame + cambio de posición



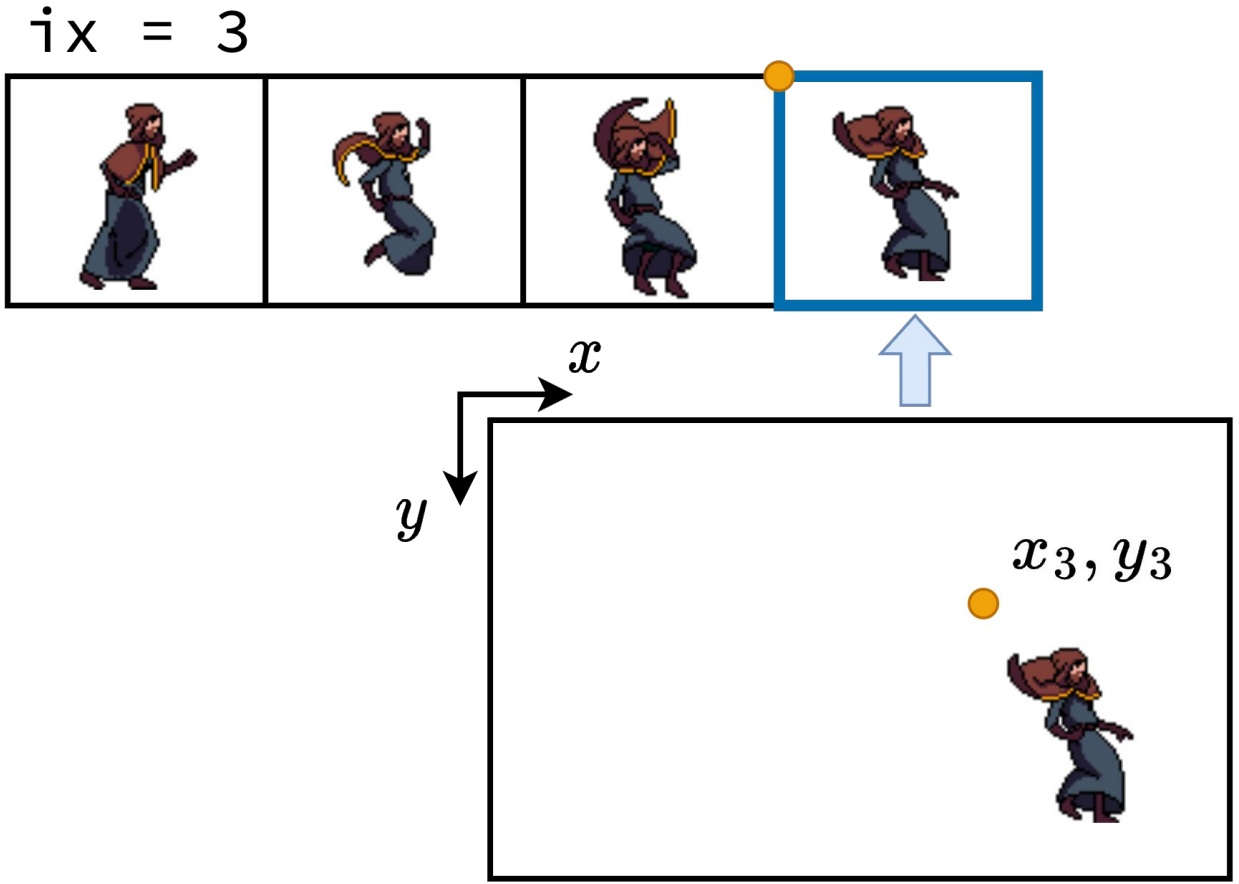
Movimiento: cambio de frame + cambio de posición



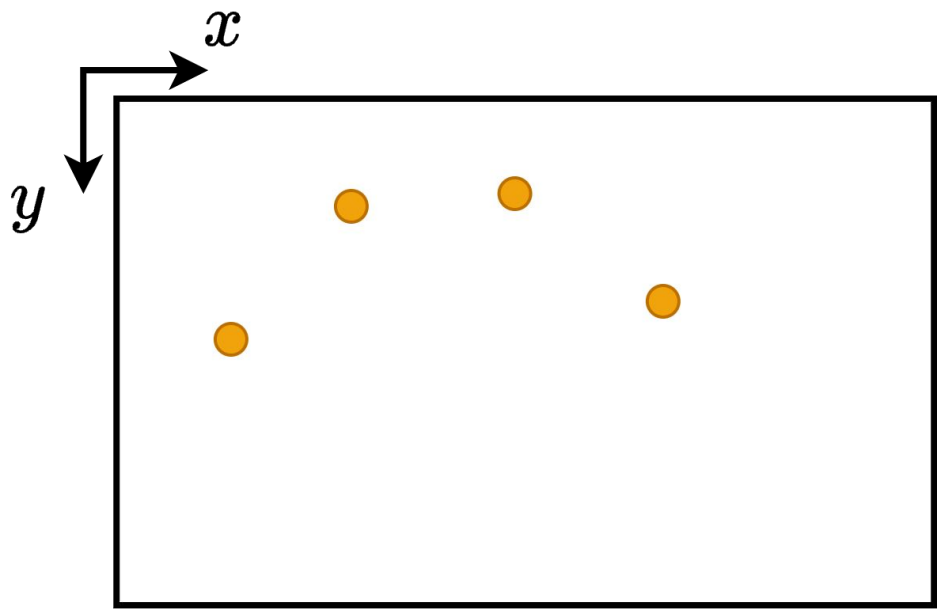
Movimiento: cambio de frame + cambio de posición



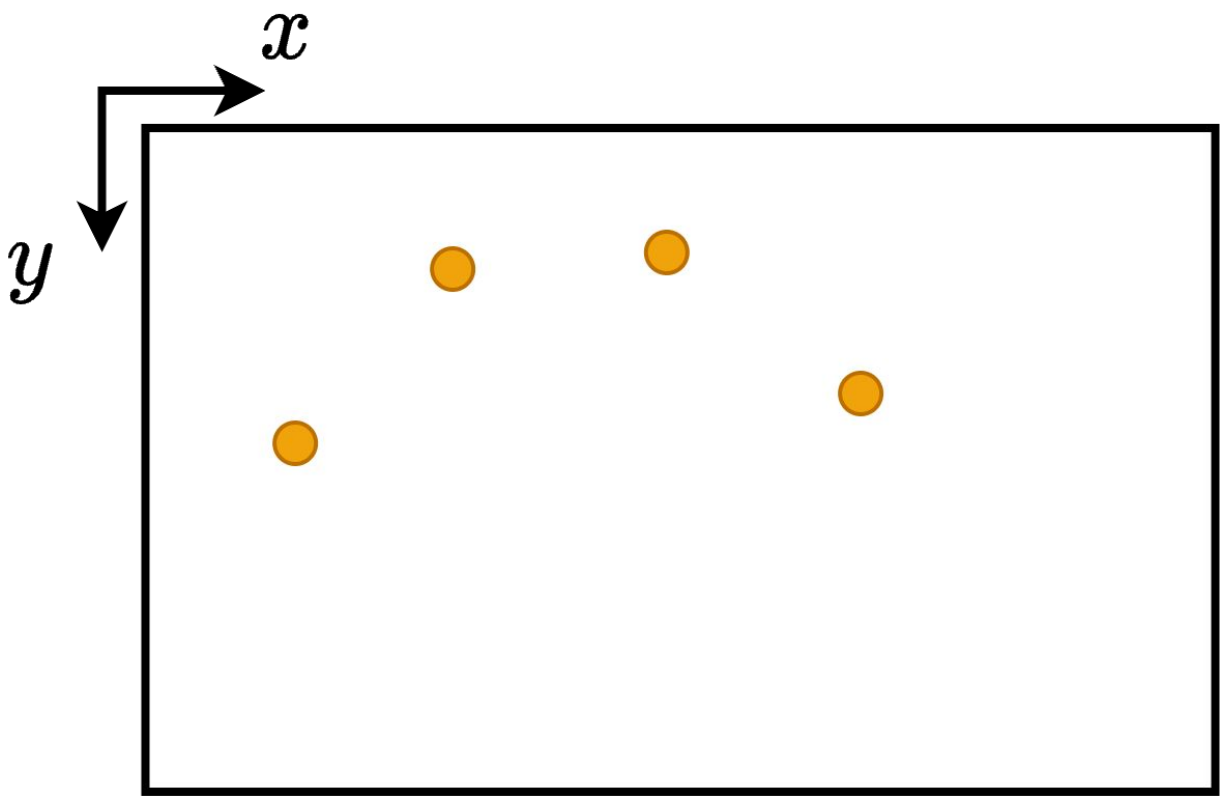
Movimiento: cambio de frame + cambio de posición



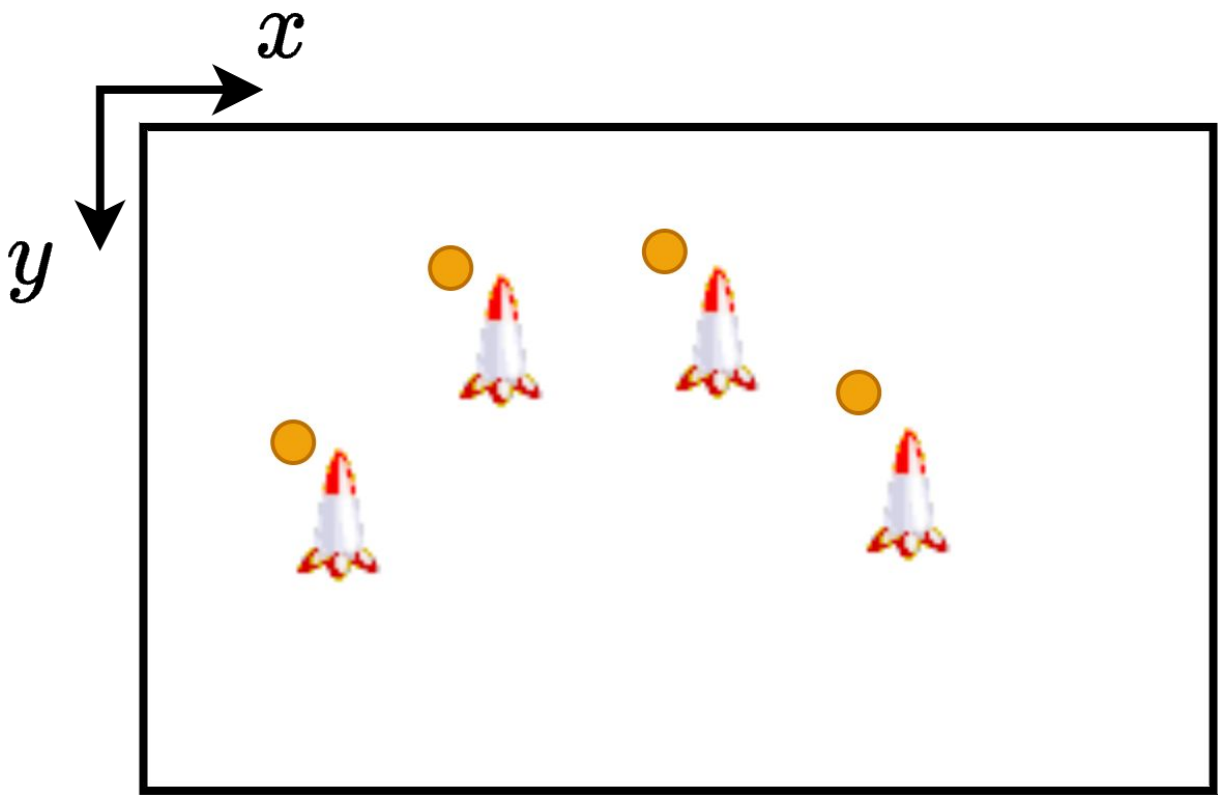
Movimiento: cambio de frame + cambio de posición


x
y

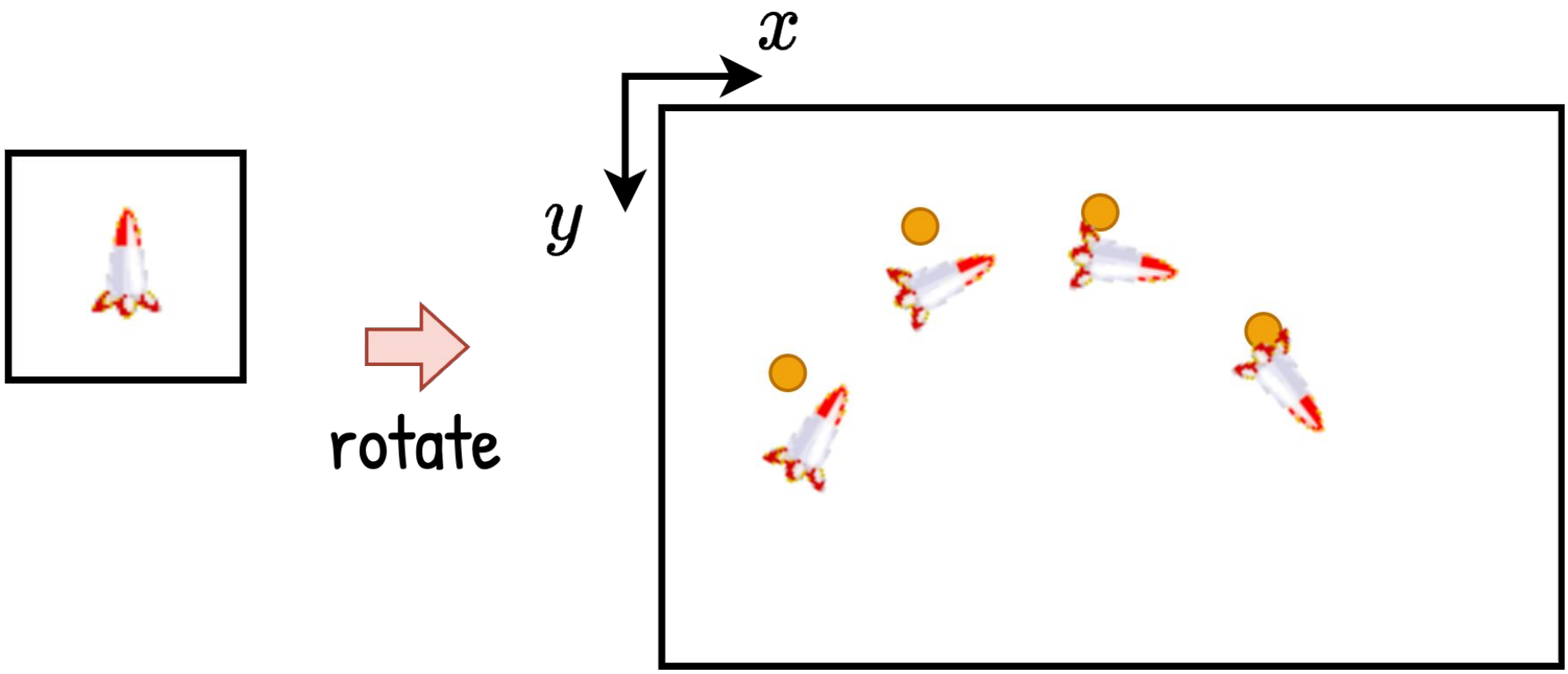
Alcanza con solo cambiar la posición?



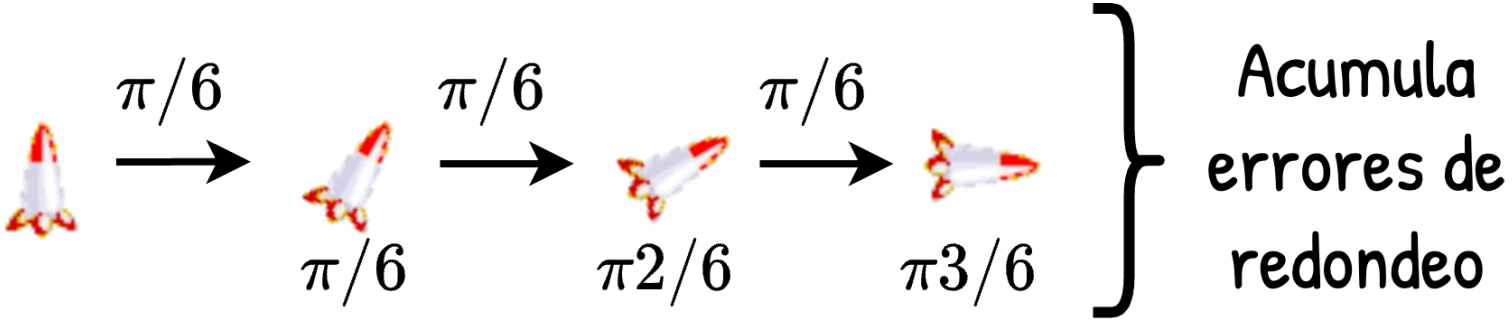
Incorrecto

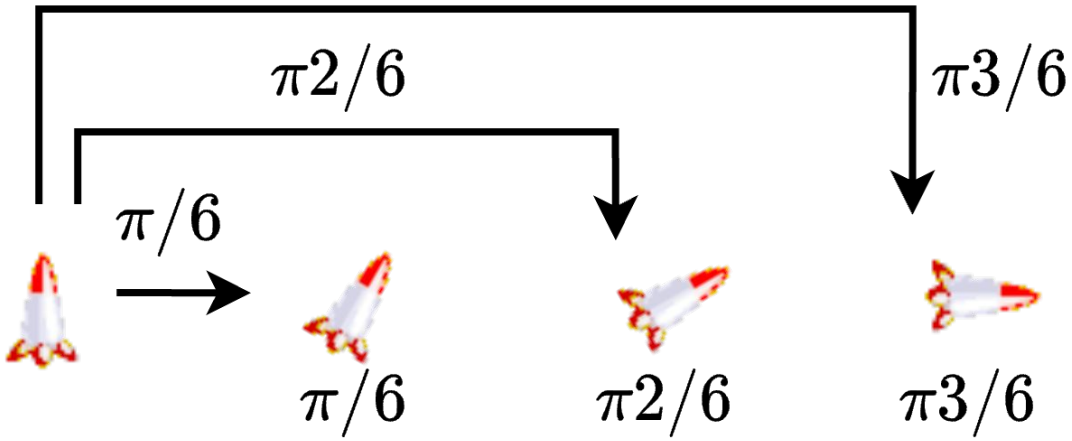


Rotación

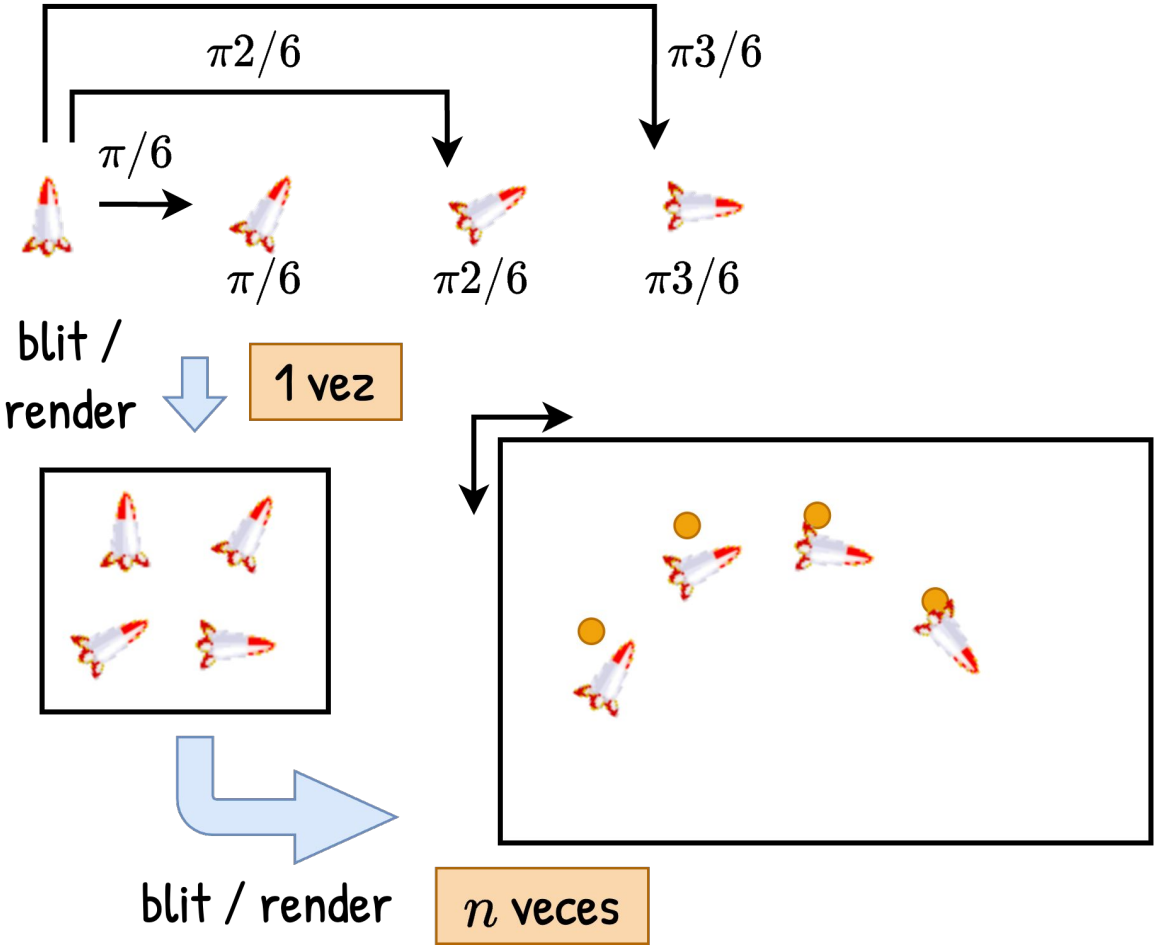


Error numerico

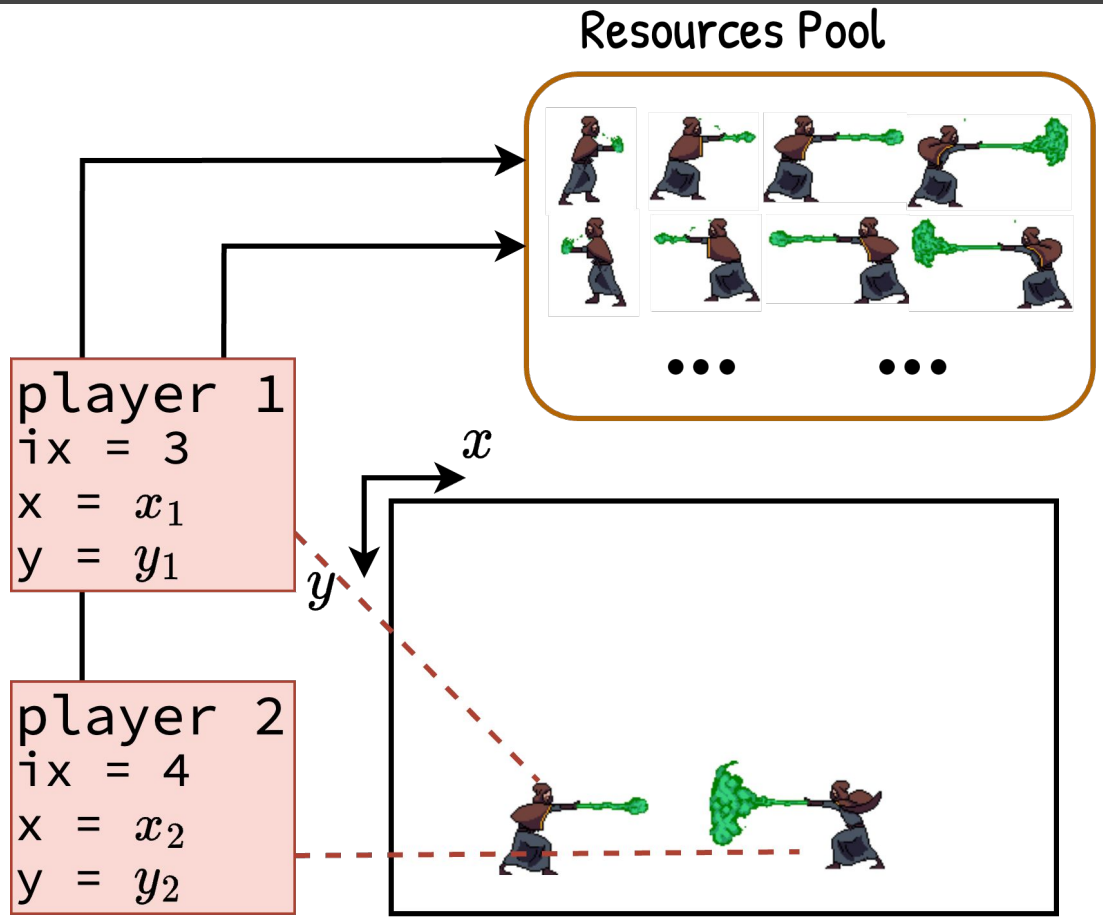




Pre-renderizado



Resource Pool



Main loop (II)

```
void main_loop() {
  init_resources();
  while (not exit) {
    update_logic_and_physics();
    clear_display();
    update_animation_frames(it);
    render_in_z_order();
    it = sleep_and_calc_next_it(FPS, x);
  }
}
```

Cliente grafico, logica en el server.

```
void client_loop() {
  init_resources();
  while (not exit) {
    update_state_from_server();
    clear_display();
    update_animation_frames(it);
    render_in_z_order();
    it = sleep_and_calc_next_it(FPS, x);
  }
}
```

Cliente grafico, logica en el server.

```
void server_game_loop() {
  init_resources();
  while (not exit) {
    receive_updates_from_clients();
    update_game_logic_and_physics();
    broadcast_updates_to_clients();
    it = sleep_and_calc_next_it(FPS, x);
  }
}
```