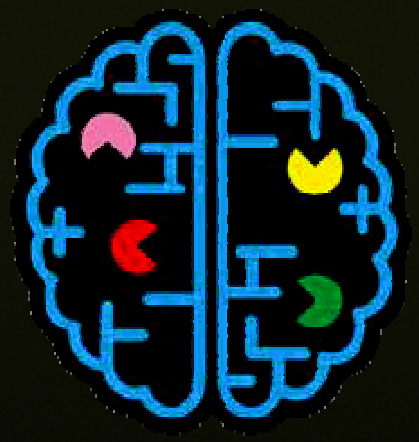

Adobe Flash and Air -
Mobile Games Fast!
Richard J. Seis
Mobile Developer Technologies
NVIDIA.

So . . . much . . . information . . .

- Conferences are long!
- Perfect – you're all of mixed disciplines.

- Tests we've done . . .
  - Definite slant toward asset creation.
  - Tests run with many games.

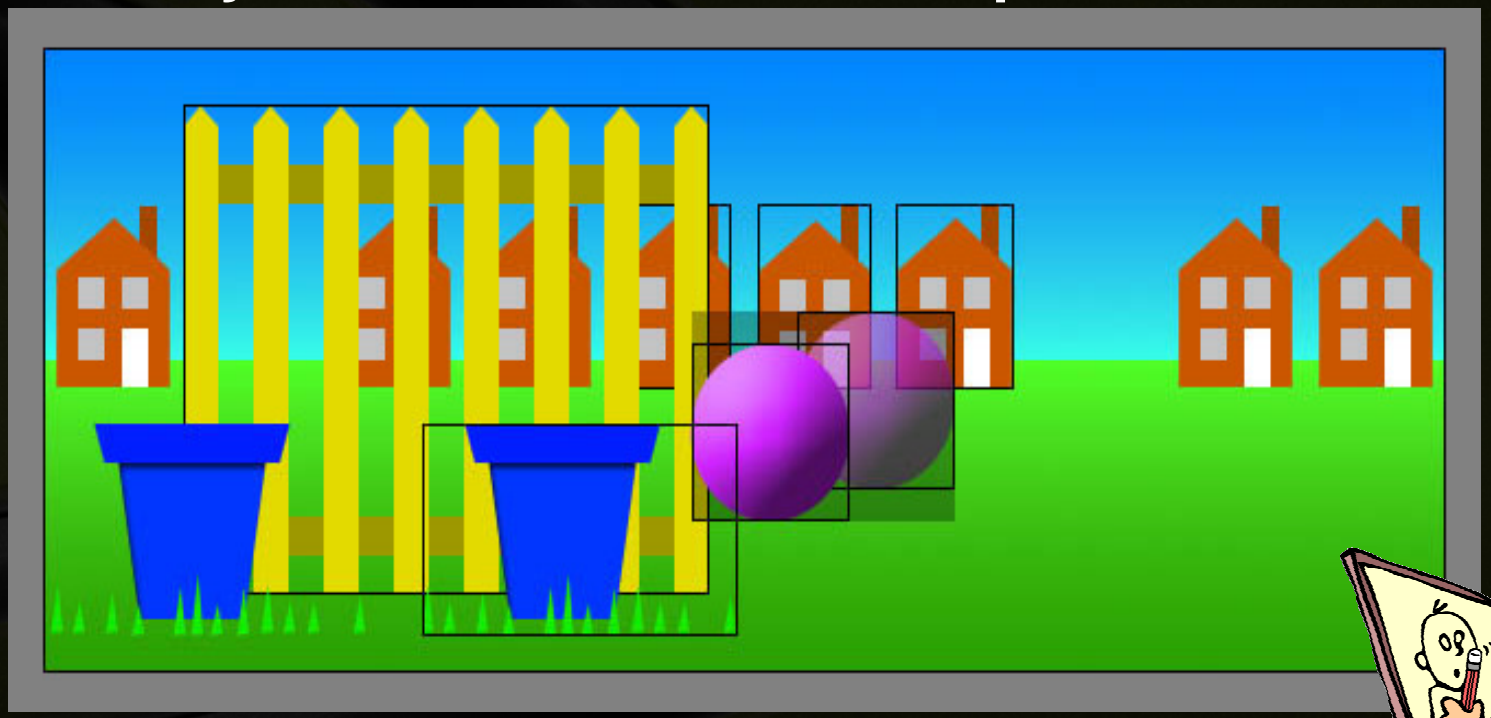
- You've heard much of this already!?
- Let's try to retain all this information!

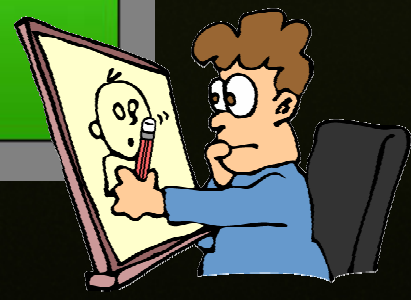


Number 1 – What's it doing

- Flash only renders parts that have changed
- Significant improvement, unless scrolling

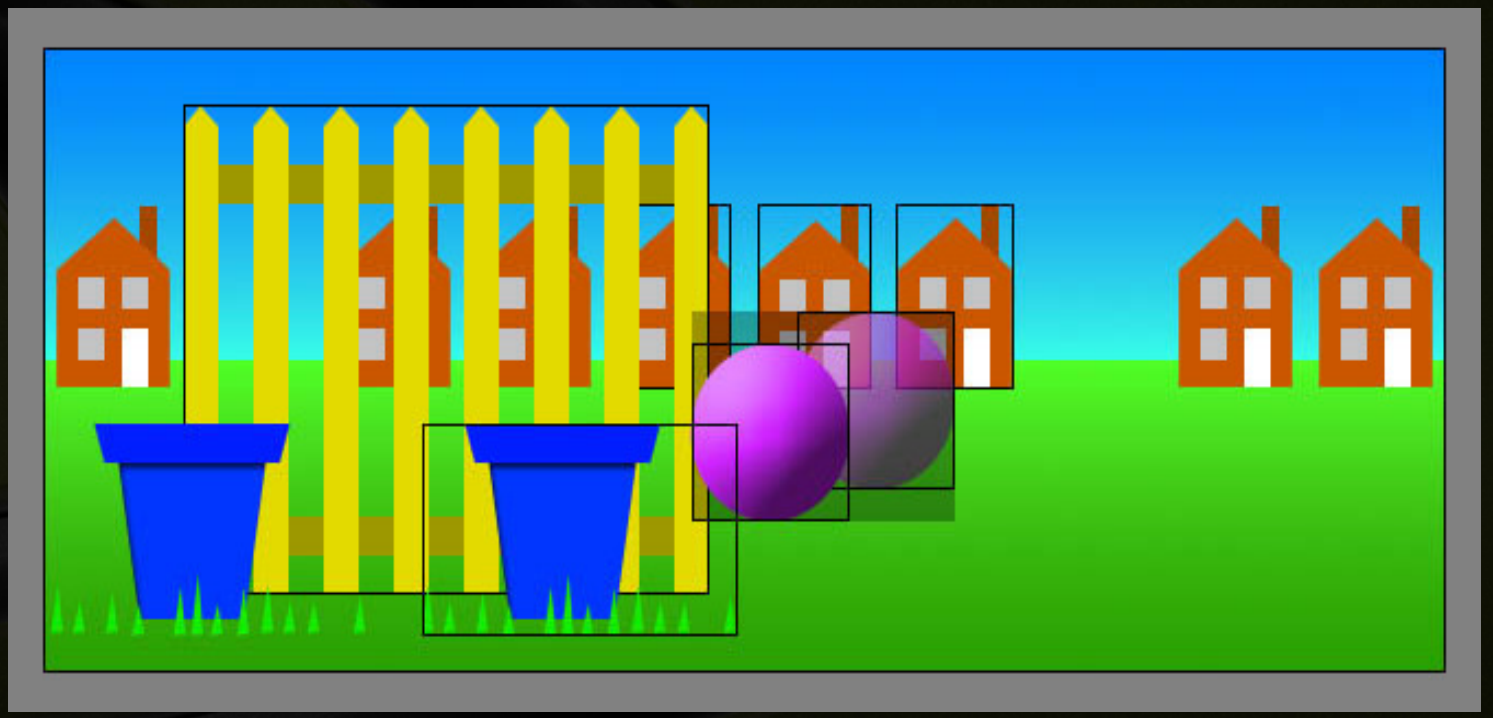
- Let's try to understand all the implications





Number 1 – Continued

- Purple ball has moved
- Redraws everything in the “dirty rectangle”
  - Ground, sky, 3 houses, fence, ball, and pots
  - Uses the bounding box, not pixels

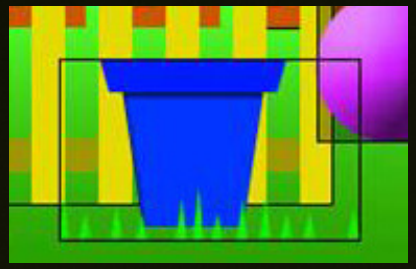


Number 1 – Continued

- What about multiple objects moving?
- Observed governing factors for dirty rectangles
  - No rectangles will overlap
  - Only 3 or less rectangles
  - Minimize the number of rectangles
  - Minimize the area that is covered
- Dirty rectangles are hard to manage in a constant scrolling game

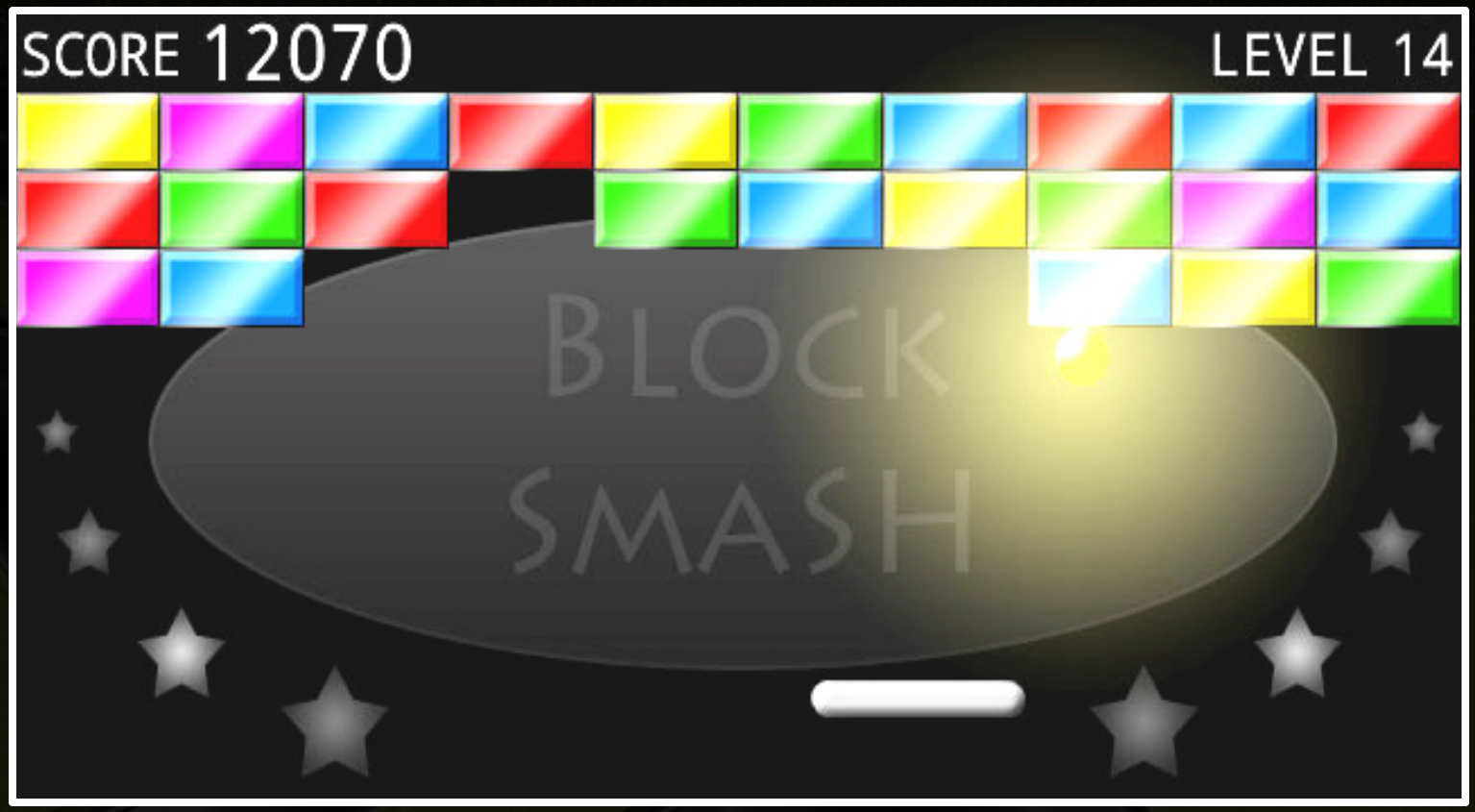
Number 2 – Is that really necessary

- Minimize what's drawn
  - Hard to when large areas are constantly changing
  - Most games, even scrolling ones, have times of pause
- For times when only parts of the scene changes
  - Avoid unnecessary change
    - Subtle animation effects
    - Large number of animating objects
    - Minor animations that contribute little



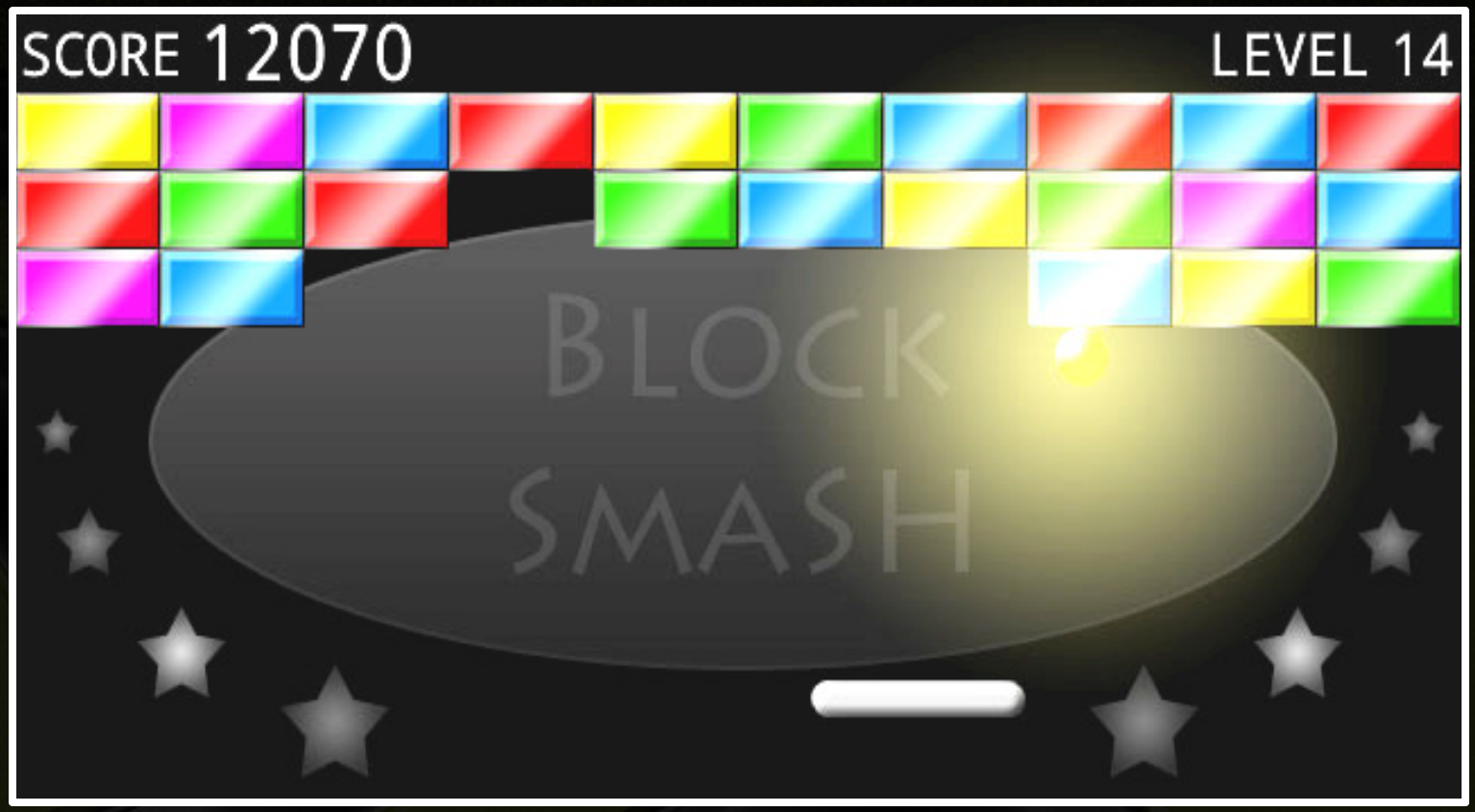
  - Avoid scattering animations, 3 rectangle limit
  - Make objects compact, reduces overlap
- Let's put this in context, introducing "Block Smash"

Number 2 – Continued



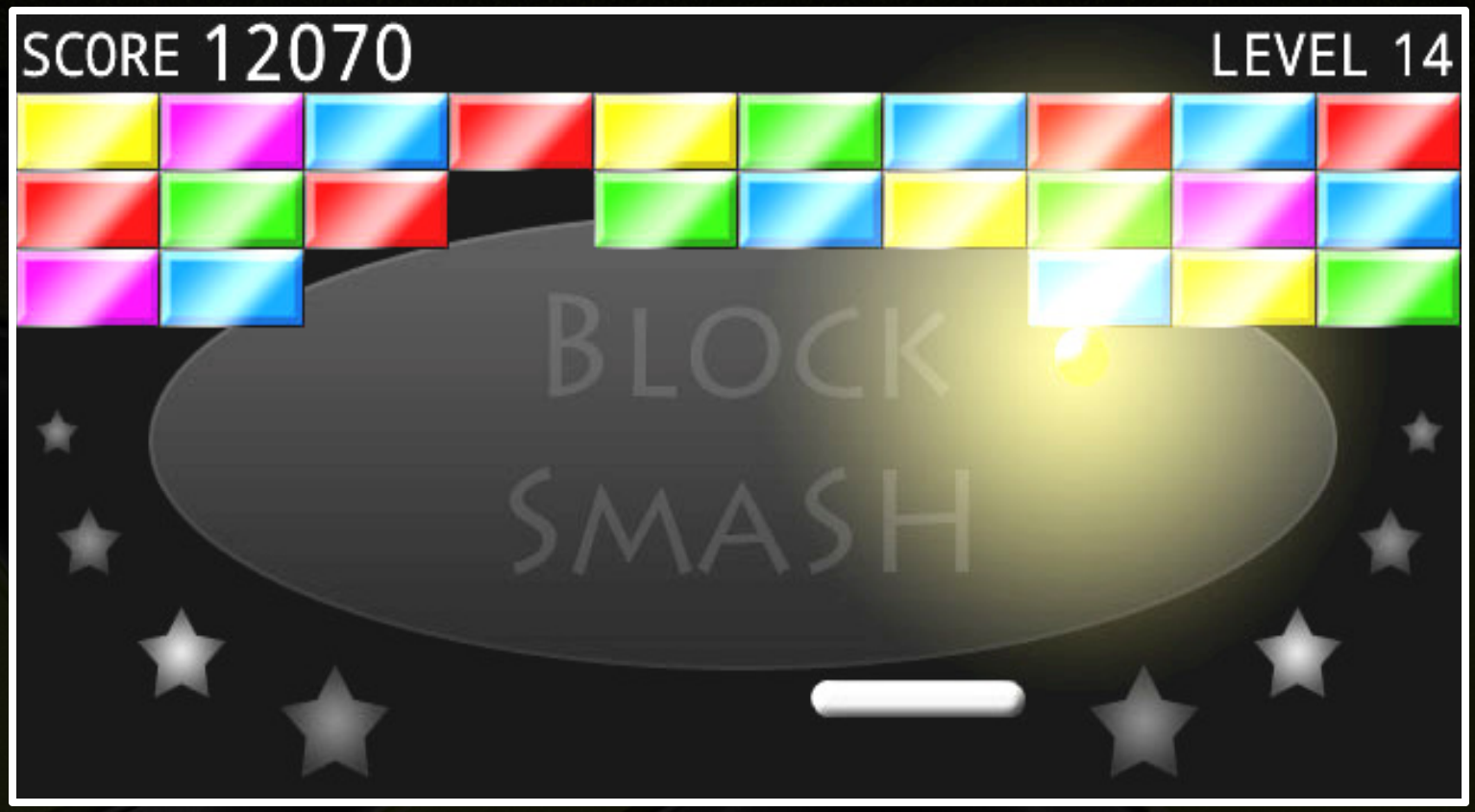
- Scattering animations – diagonal reflection is played on all of the blocks simultaneously when any is hit
- Try having an animation play sequentially one column at a time

Number 2 – Continued



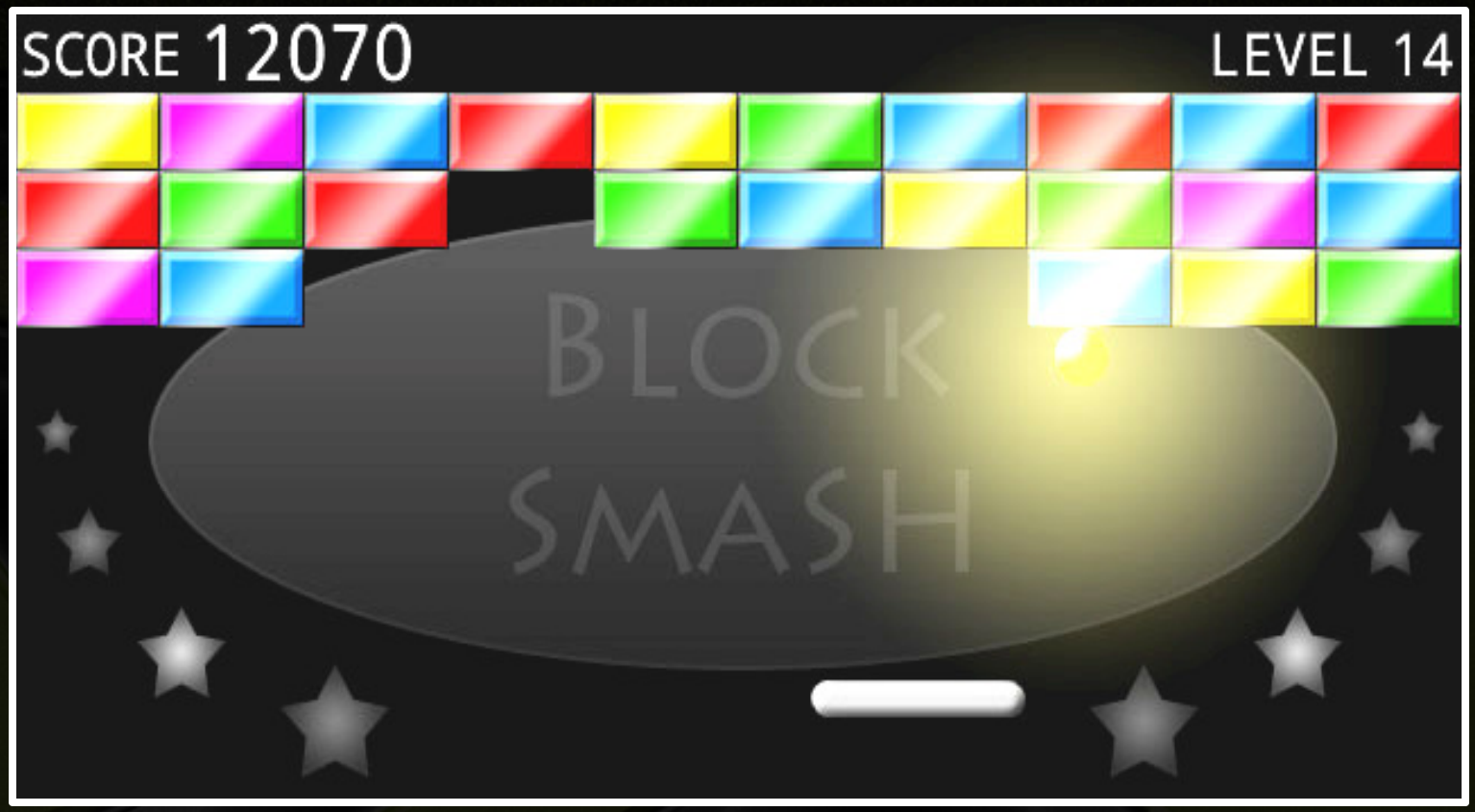
- Subtle animated effects that cover large areas – large glow around our ball
- Try simply making the ball or effect smaller

Number 2 – Continued



- Avoid animations that contribute little – all the stars have a constant twinkle animation
- Try making them static or stagger animations

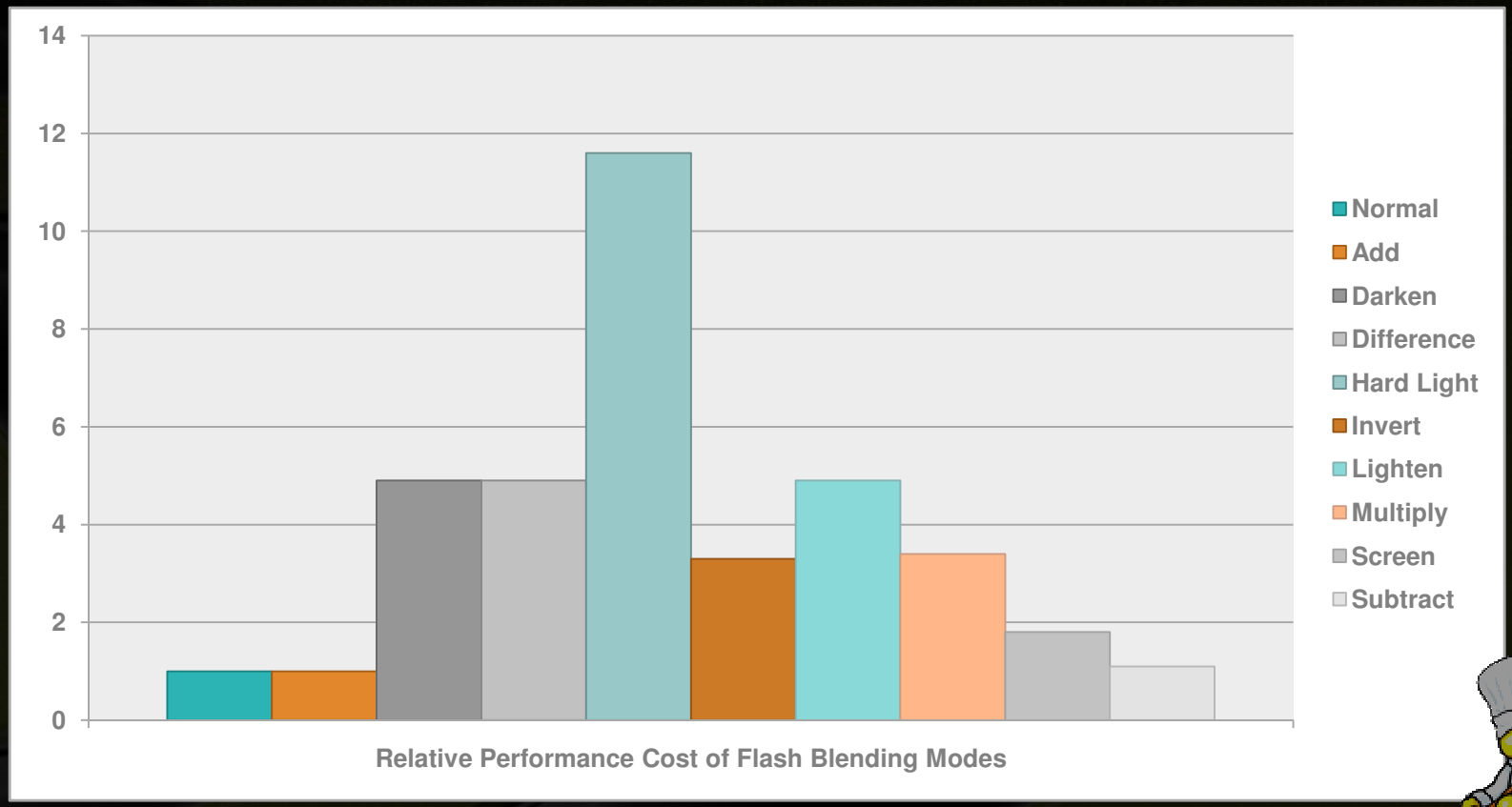
Number 2 – Continued



- Keep intermittent animations short – the score text has a scale animation applied when it changes
- Try making it shorter, eliminating it, other type of indicator

Number 3 – Careful when you mix

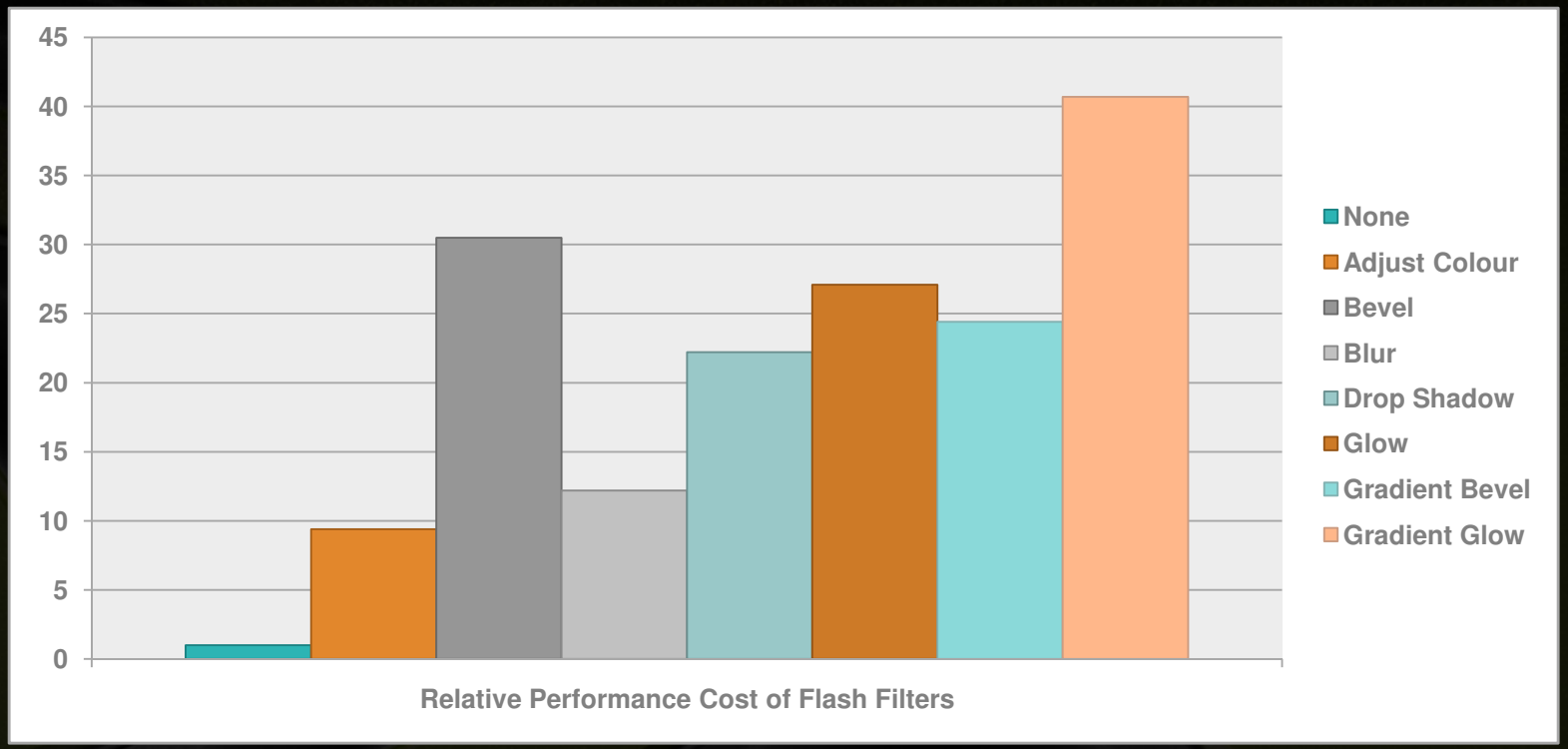
- Blending modes vary in performance



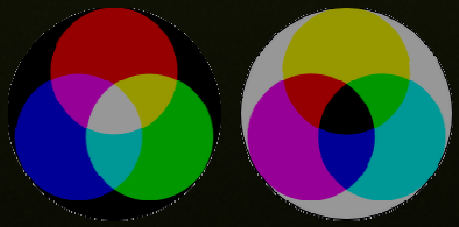
- Slowest – Hard Light – 12X
- Fastest – Add, Screen, Subtract – < 2X

Number 4 – Be who you are

- Filters can be more expensive than blending

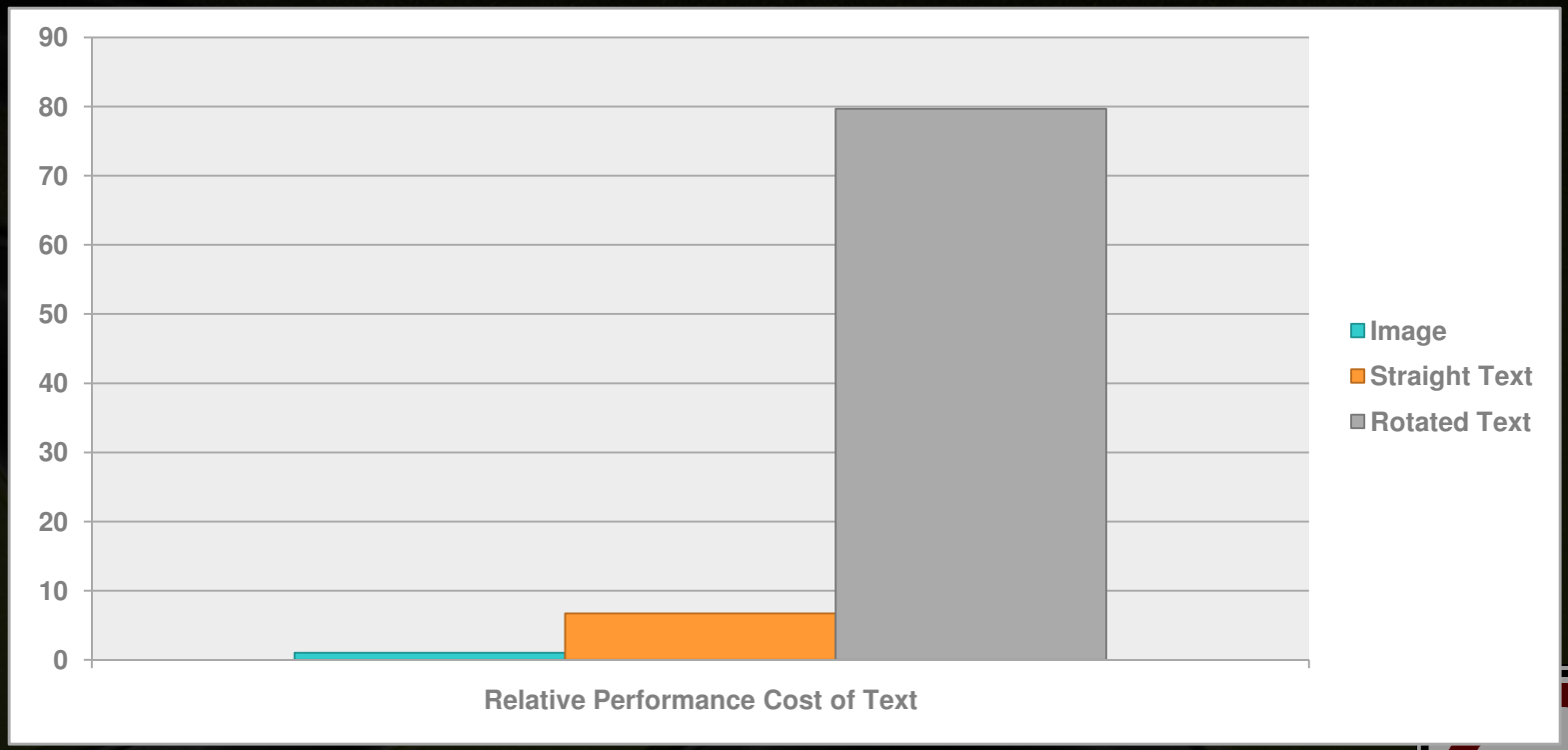


- Cheapest – Adjust Colour – 9X
- Most Expensive – Gradient Glow – 40X
- Maybe “bake” it in



Number 5 - Don’t write, show me

- Text is expensive, but is needed



- Make text that doesn’t change an image
- Straight text is 6X
- Rotated text is 80X

Number 6 - We're better in groups

- Reduce the number of draw calls
- "Draw call" – rendering of similar object(s)
  - There is setup time per call
  - Generally, reducing the number, increases performance
  - Ideally in the low hundreds per frame
- Reduction Methods
  - Set objects to use cache as bitmap where practical
  - Combine as many of the object shapes as possible
  - Where there's animation, separate the static from animated

Number 6 - Continued

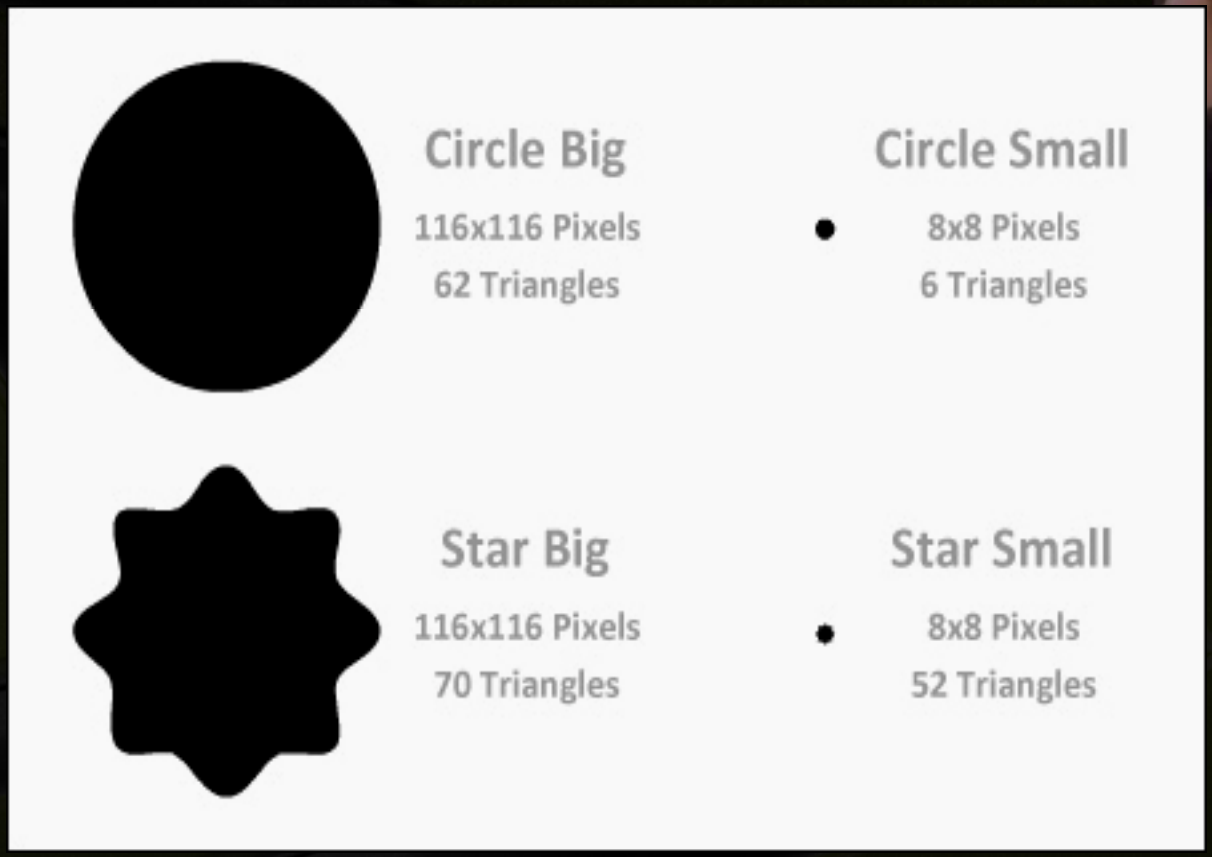
- Example of separating the static from animated

- Separated into two objects
  - Static object – the birdhouse
  - Animated object – the bird

- The birdhouse is reduce to one draw call as it can be cached as bitmap
- The bird is the only shape potentially needing multiple draw calls

Number 7 - Keep it simple

- Reduce shape complexity
- GLES2 tries . . .



- If you really need it – maybe do LOD

Number 8 - The right way to hide

- Simple but important
- How to hide an object

- BAD – set its Alpha value to 0
  - Hardware still does all the work
  - Even if you fade, use below when finished

- GOOD – set its visible property to “false”
  - No cost to the hardware
  - Turns rendering off completely

Number 9 - Use the tools

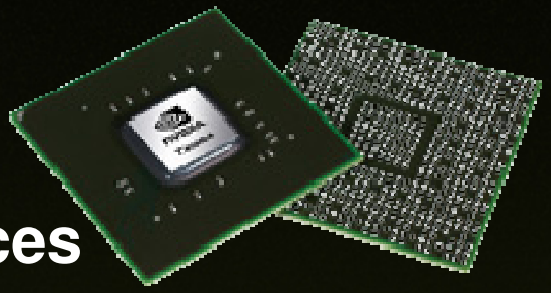
- None of these tips are helping!
- Now's the time for the profiling tool
  - You may have heard of these already
  - On Tegra platforms – PerfHUD ES
  - Matters more than ever
- PerfHUD ES – for instance
  - Performance Dashboard
    - Primitive counts, Draw calls, Batches, etc.
  - Frame Debugger
    - Frame Scrubber, Geometry Viewer, Texture Viewer

Number 10 – This is important

- No really!
- These are just some of the dials you can turn, let your game dictate needs
- Platforms are low powered, relatively
- Becoming increasingly important, mixed
- Start early
- Keep learning, testing, and helping

Conclusion

- Performance is team wide
- No single “big win,” do multiple
- Remember there are other resources



- See the Tegra Developer’s site
  - http://developer.nvidia.com/tegra
  - OS Support packs
  - SDK’s, demos, apps
  - Docs, Whitepapers
  - Development Tools
  - Public support forums
  - Access to the Tegra board store

What's Next

- Next session
  - How to – Unreal Engine for NVIDIA Tegra
- Questions
- Where else can you find us?
  - Twitter: nvidiadeveloper
  - Website: http://developer.nvidia.com
  - ***** ADD ADOBE STUFF – branding, data