Aliasing (and anti-aliasing)

Part A: understand the problem

Lecture 23 – part A

Aliasing
a fundamental topic in computer graphics

Very general problem:

continuous world -> discrete set of “observations”

finite set of pixels – each must be one color

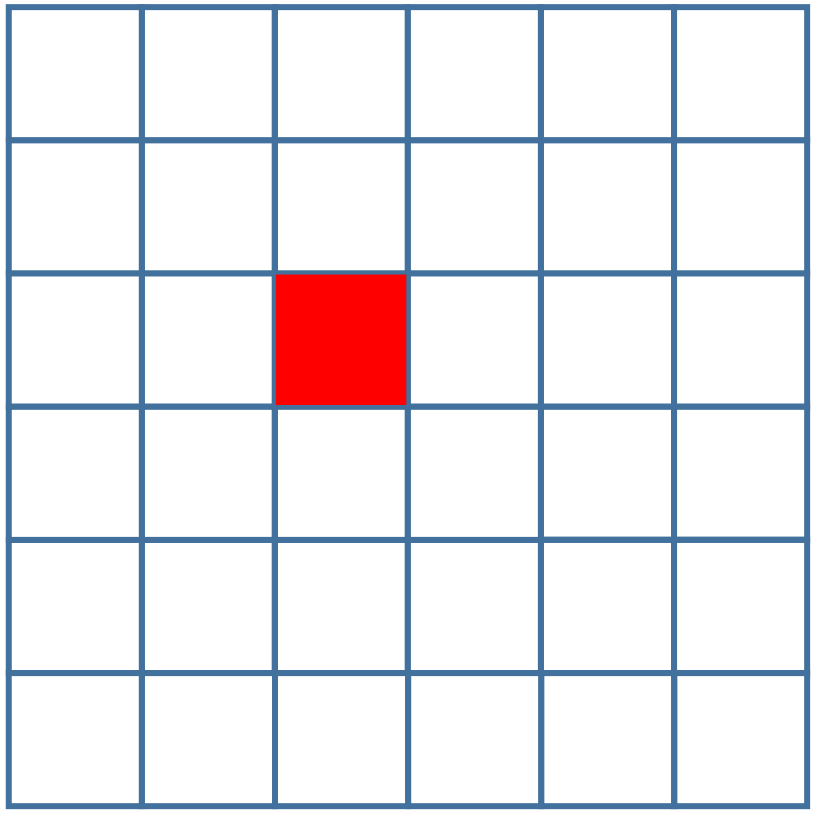
(and many other problems)

Happens outside of graphics

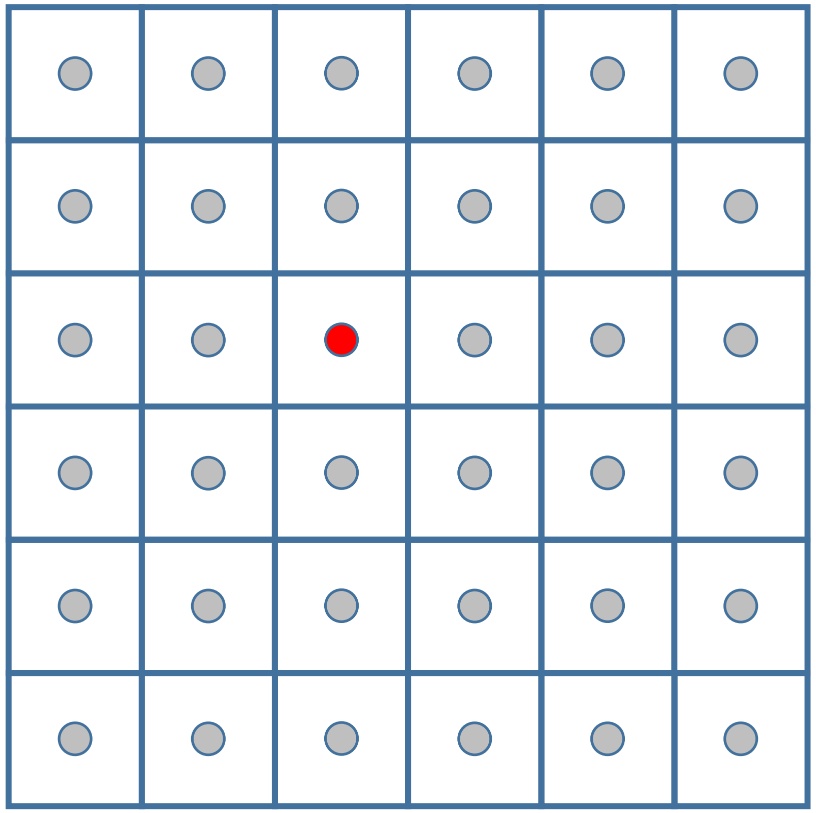
audio signals, electrical signals, …

What is a pixel anyway?

Little Square Model

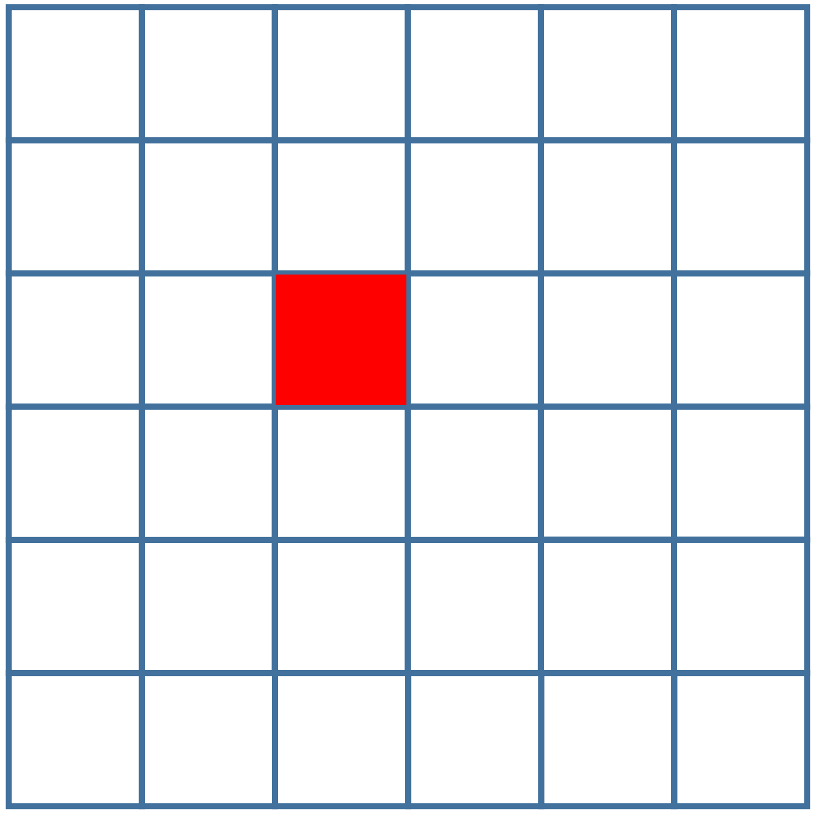


Point Sample Model



Similarities – a pixel has 1 value (color)

Little Square Model



Regions aren't really square

Hard to extend

Makes math less neat

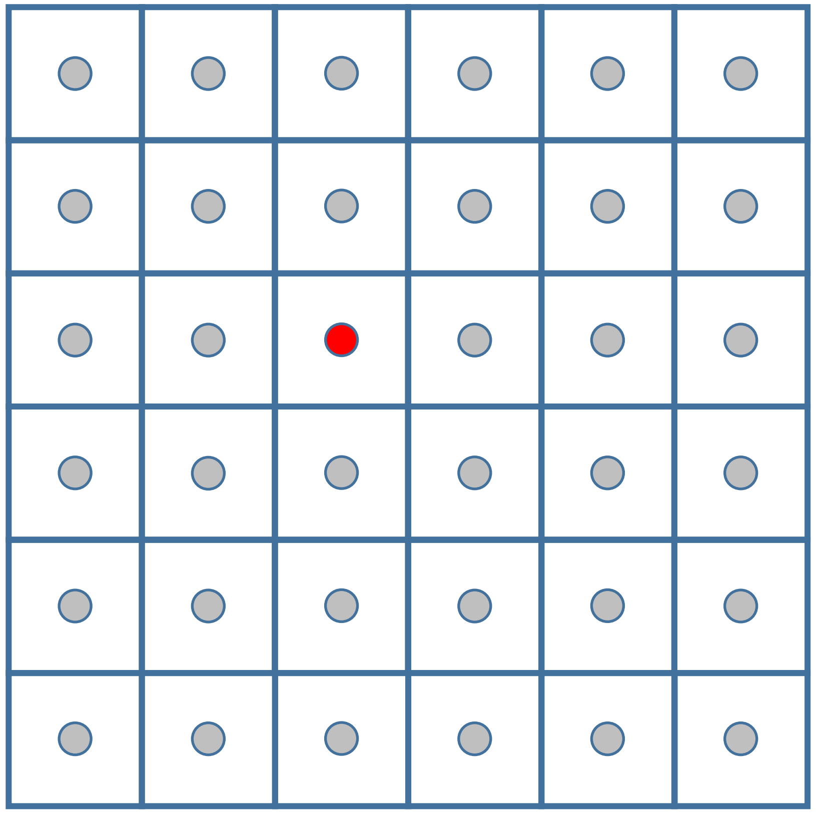
But useful for learning

Sample Points

Each sample is a specific location (no area)

There is space in between

Point Sample Model



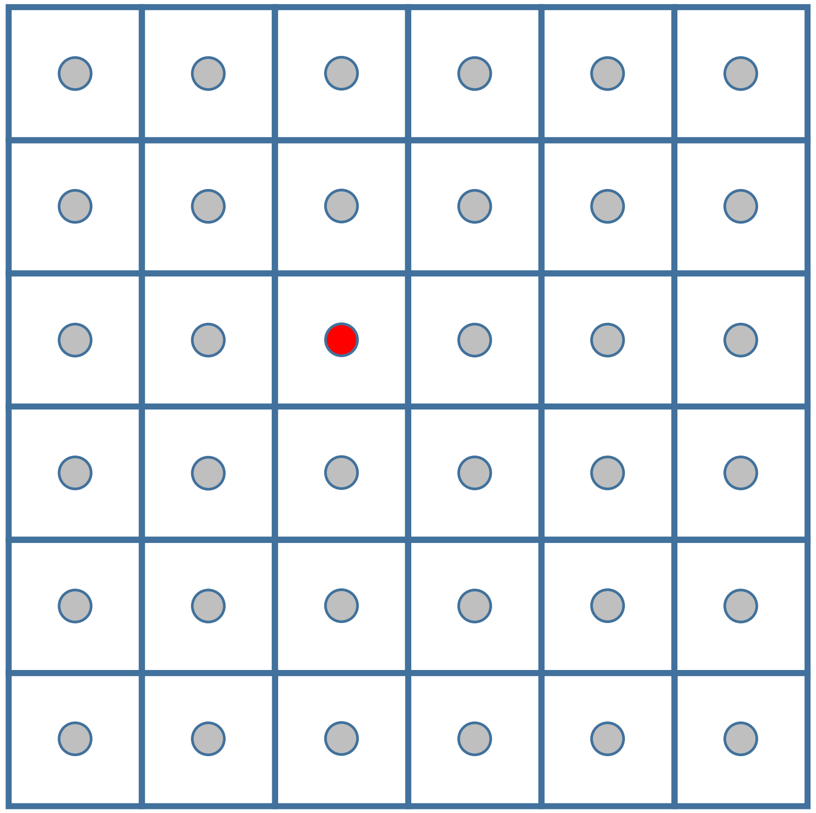
From points to squares (if you need to)

What happens in between?

Simple thing: nearest neighbor.
(leads to squares)

Different choices are possible

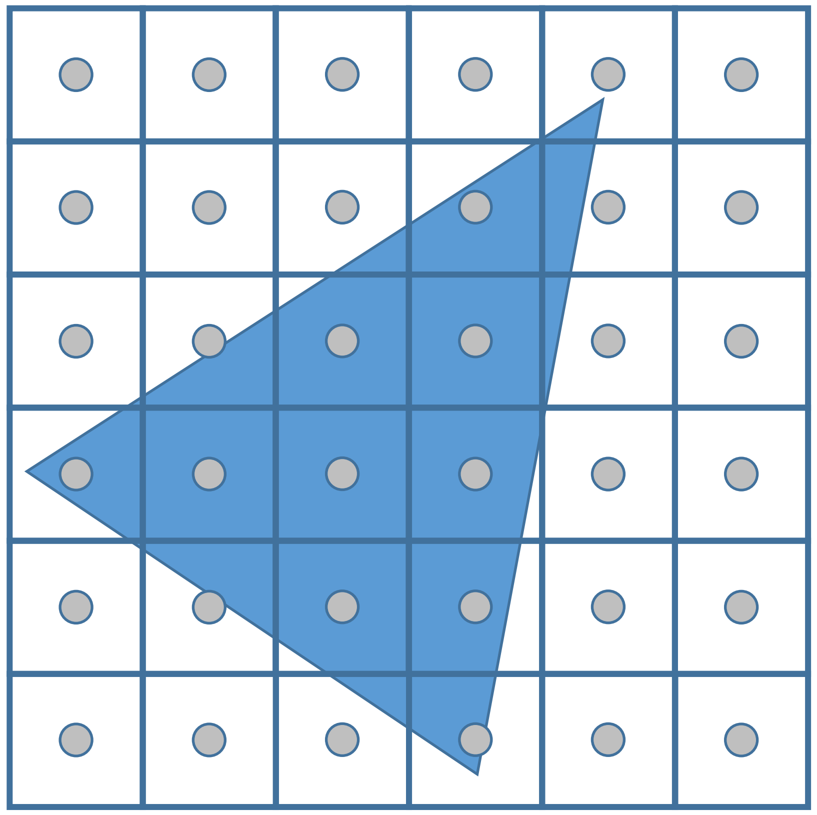
Point Sample Model



Triangles

Does the triangle cover the sample point?

(simple choice)



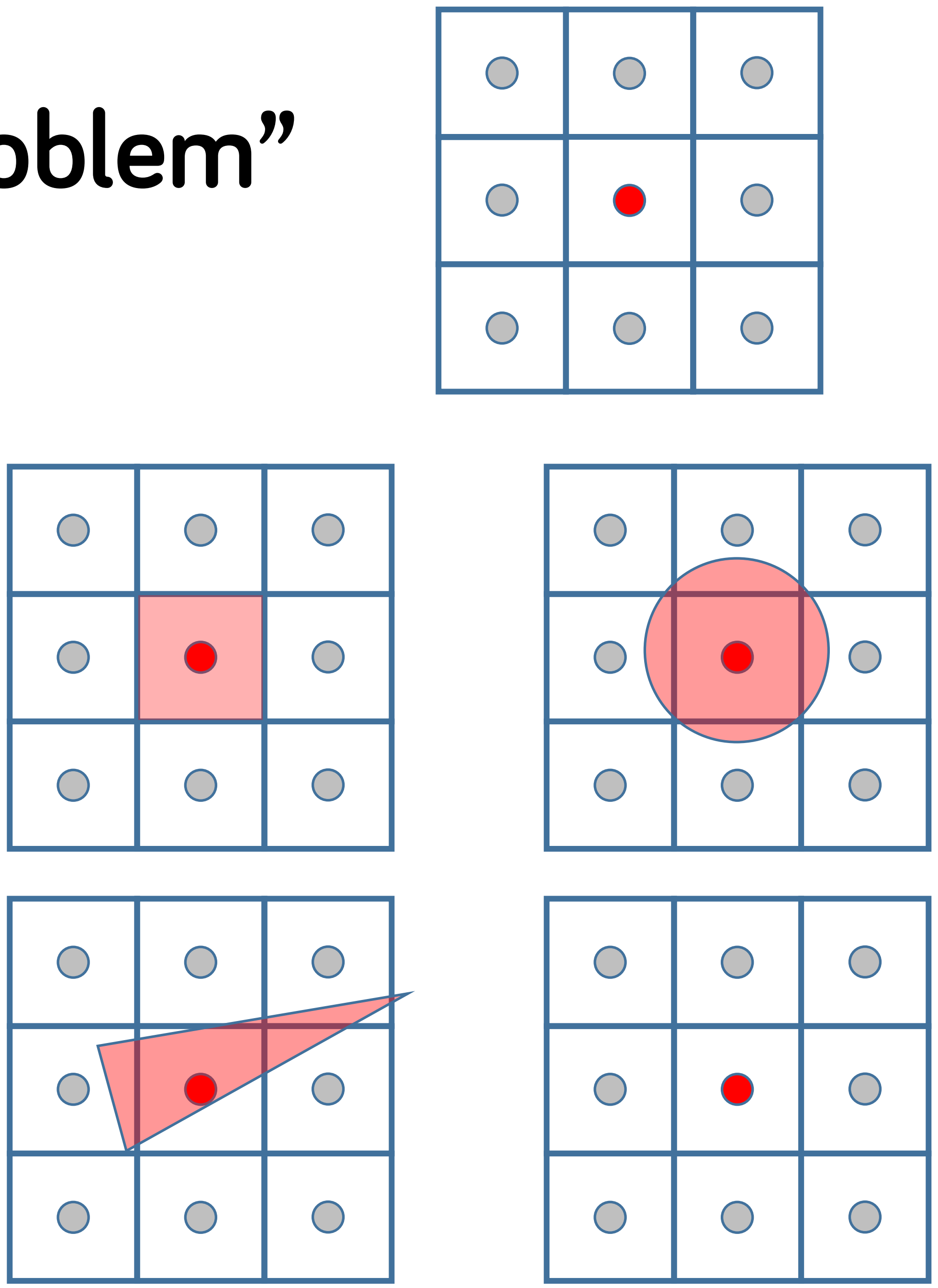
Aliasing: the “technical problem”

Discrete representation has less information

(continuous world is infinite)

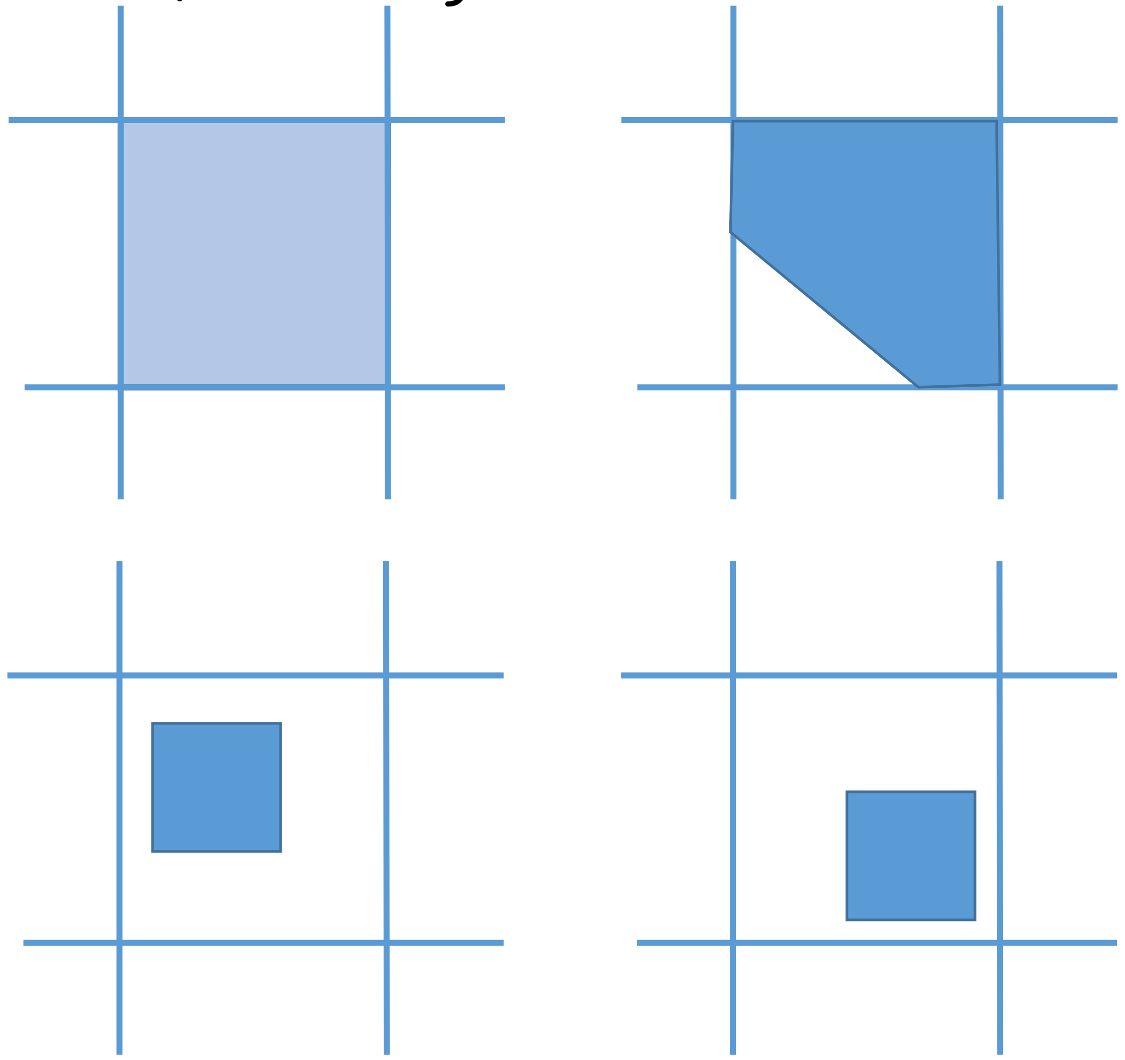
Many possible things look the same

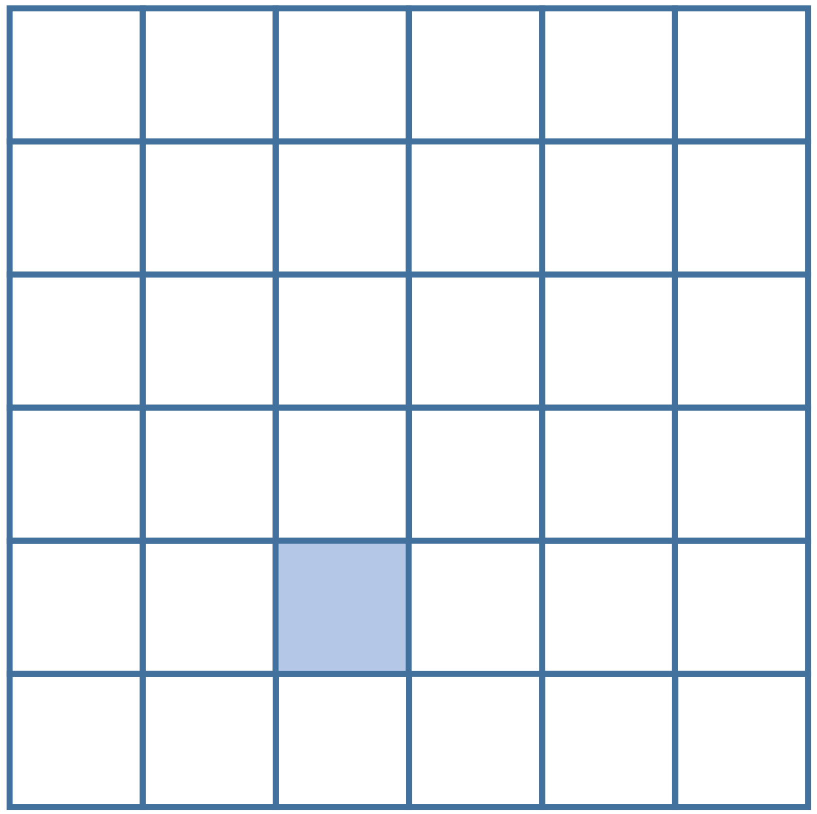
They are **aliases**



Little square model doesn't help

(actually makes it worse – less clear what it means)



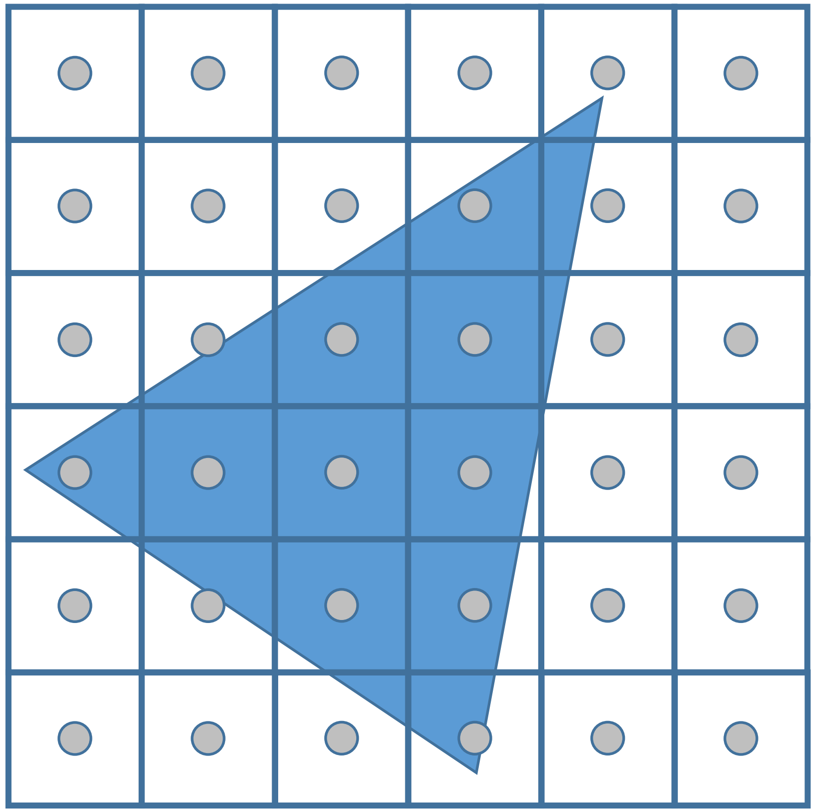


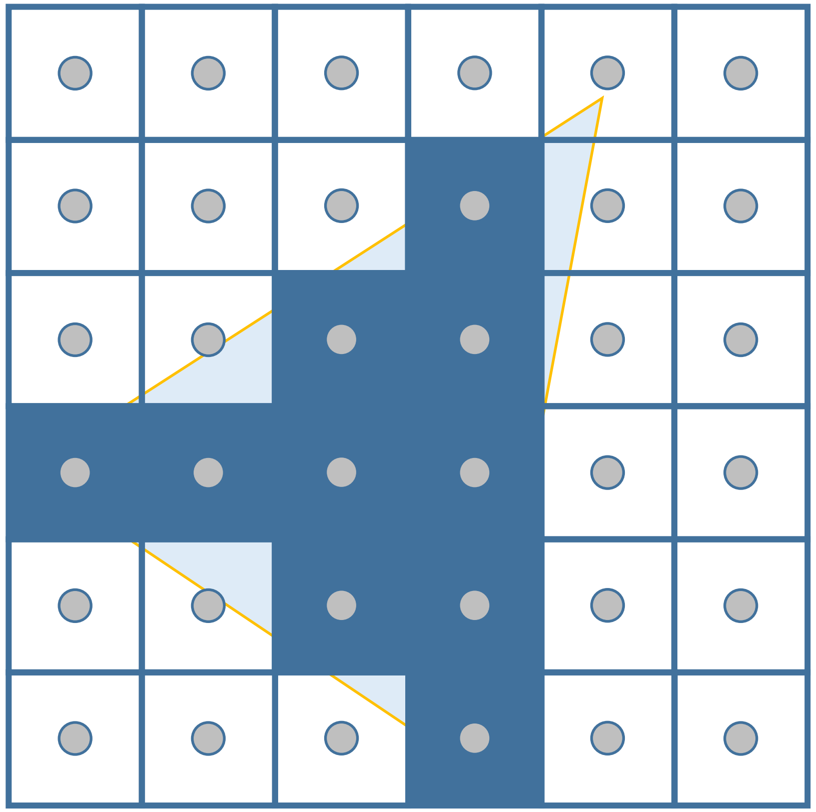
Why do we care?

Common problem any time we have a discrete representation of a continuous signal

Causes of many problems in graphics

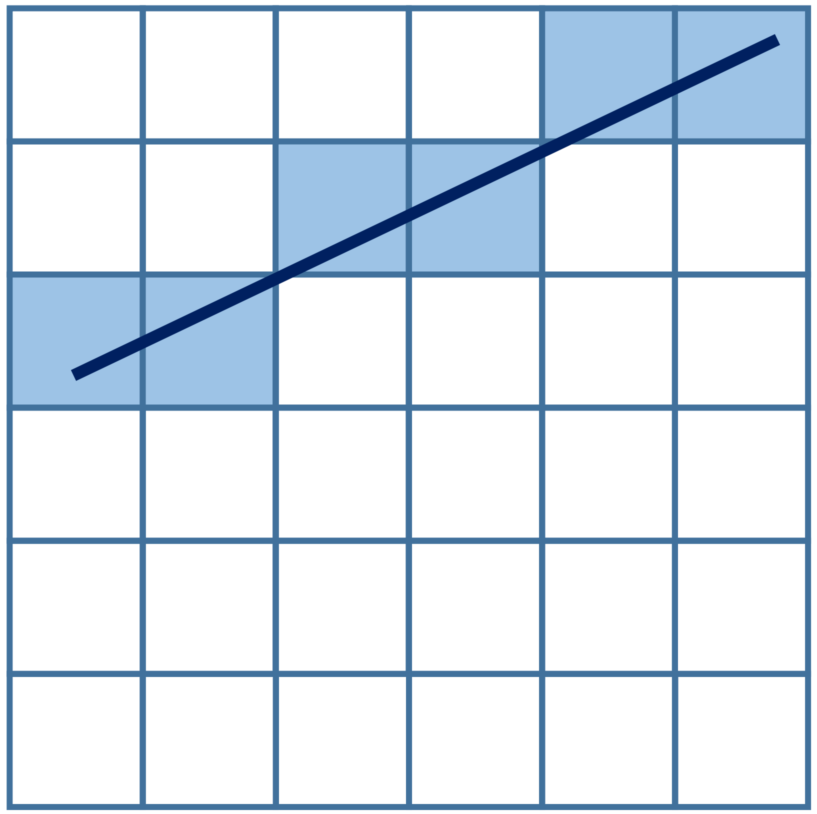
Jaggies: Inside or Outside of a Triangle





Jaggies: Lines

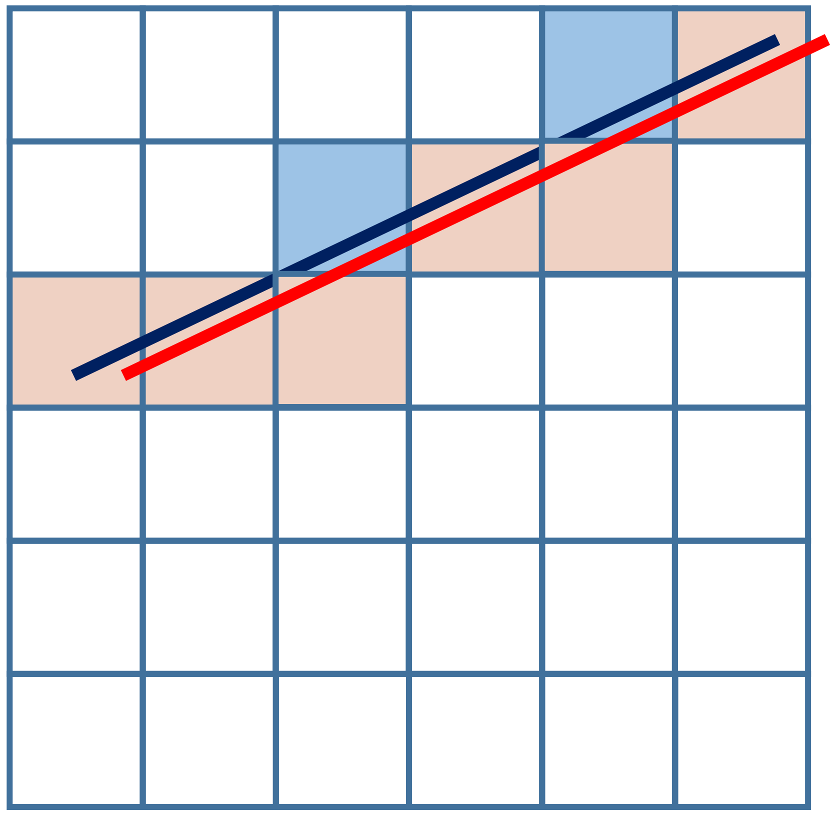
Question of which pixels to include



Lines

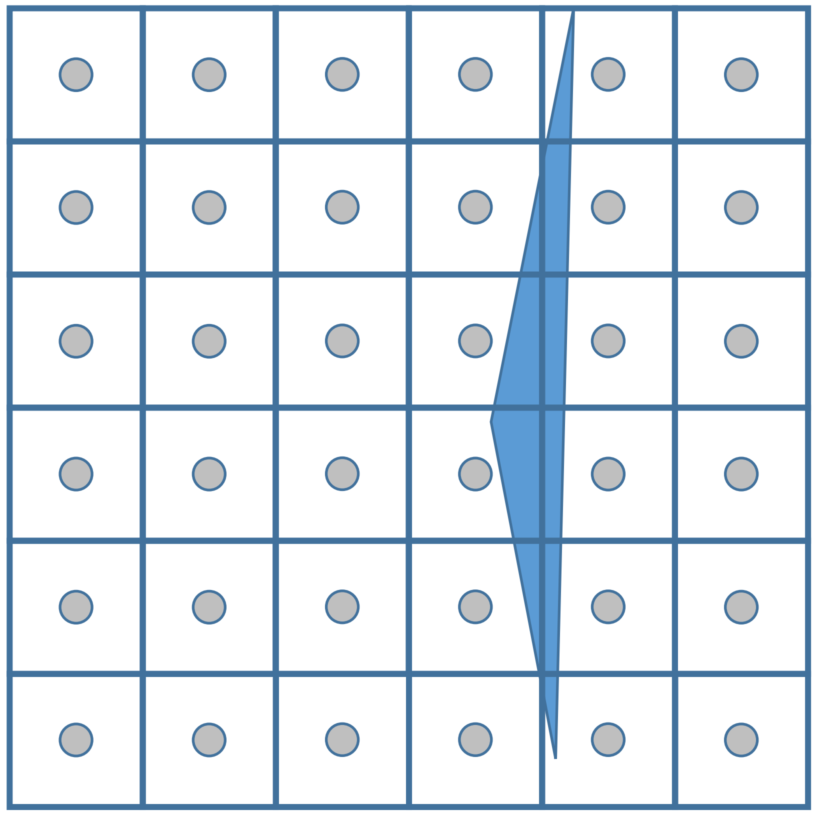
Crawlies

(mainly if edge)



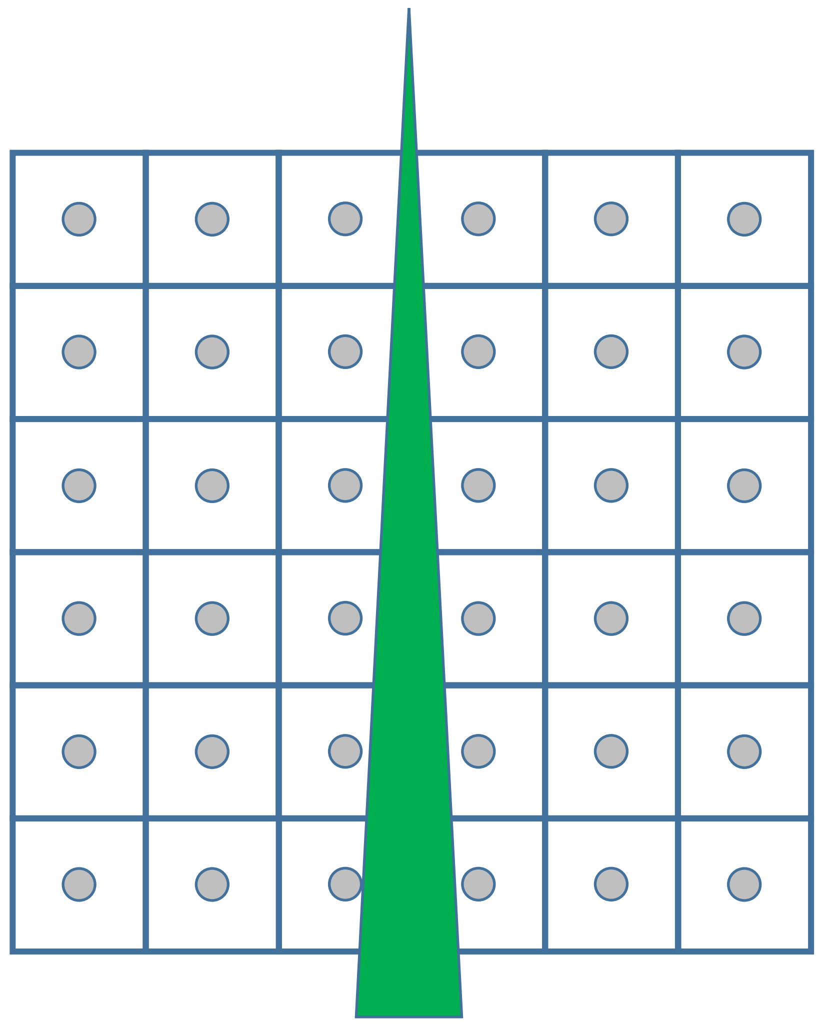
Triangles

A triangle could get lost between pixels



Triangles

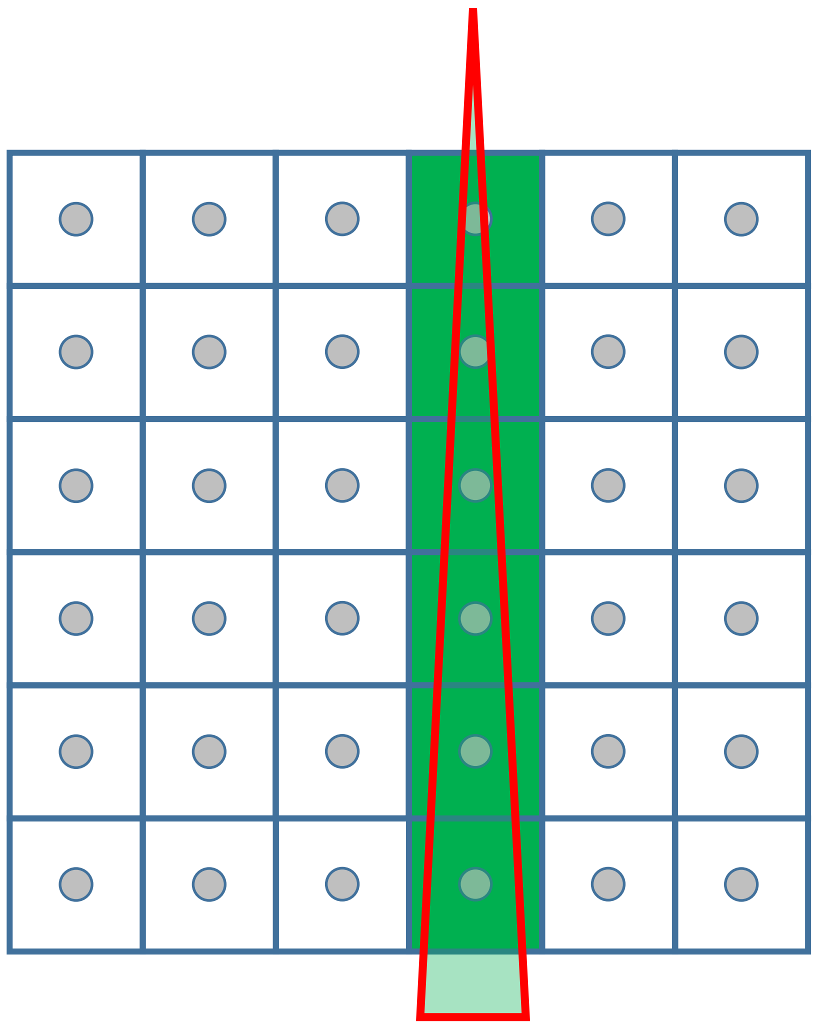
A triangle could get lost between pixels



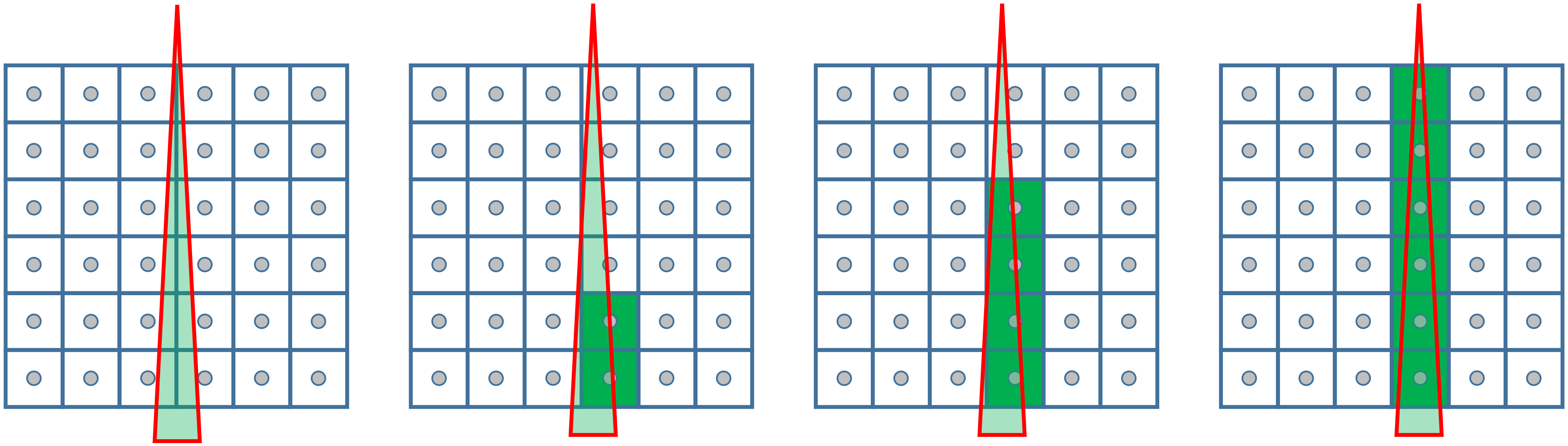
Triangles

A triangle could get lost between pixels

Or not...



A Weird Crawlie…
Small left to right motion = big up and down

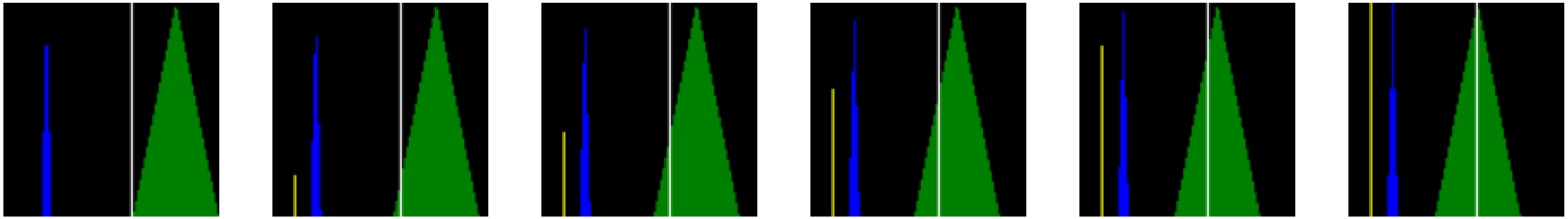


Yes – this can really happen

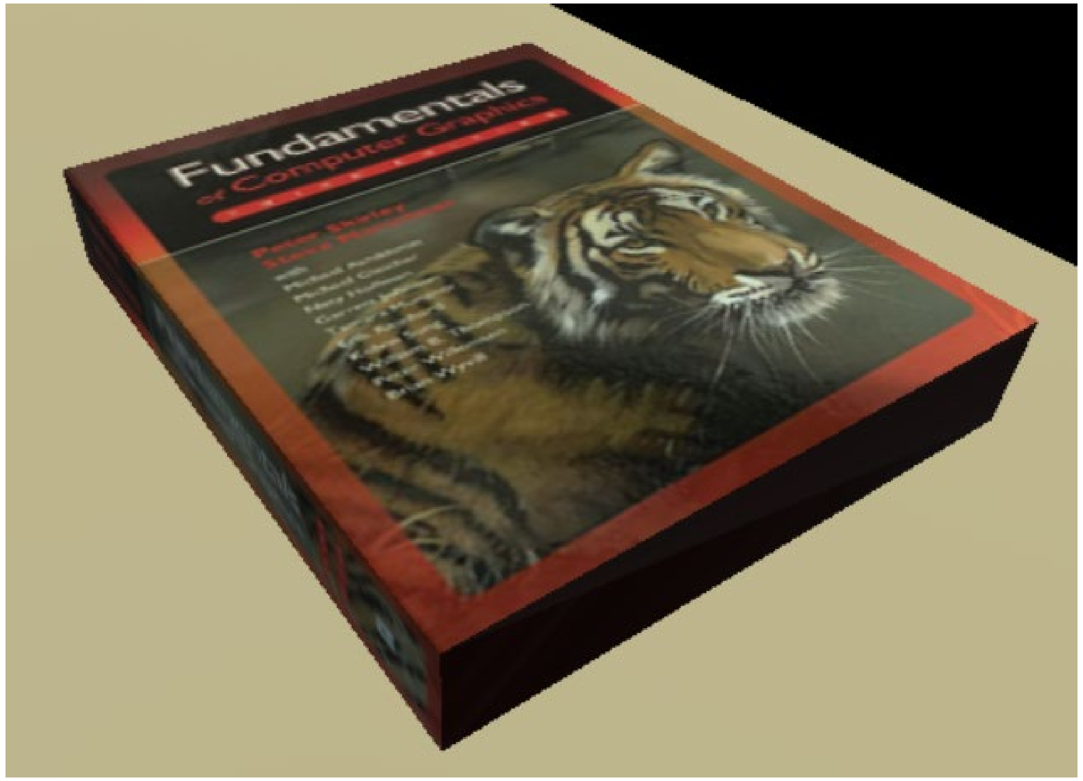
Demo

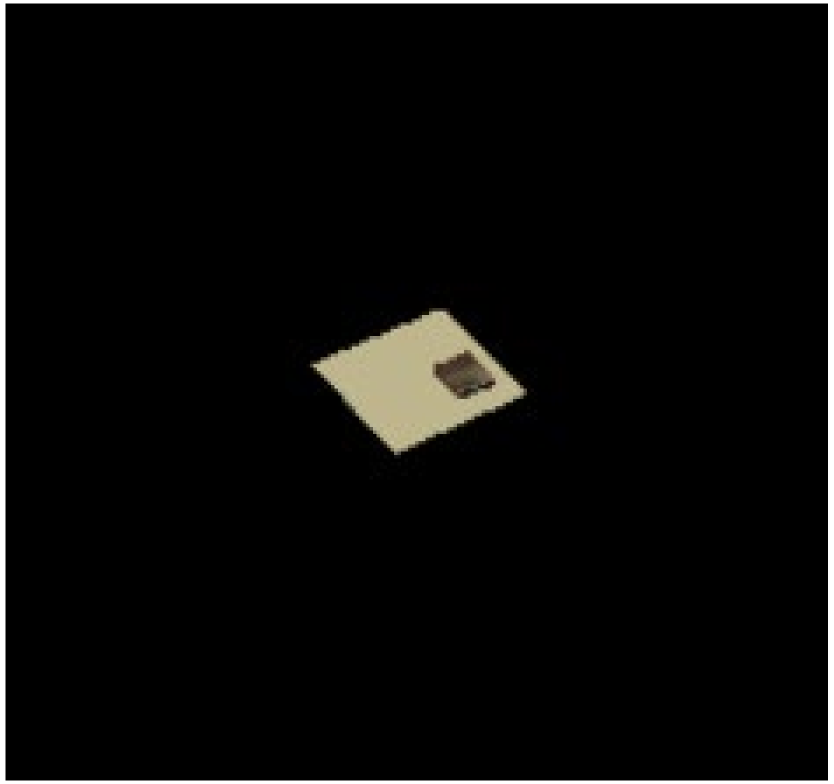
Yellow triangle is 1 pixel wide

(green is a “zoomed in version” – white is center of pixel)



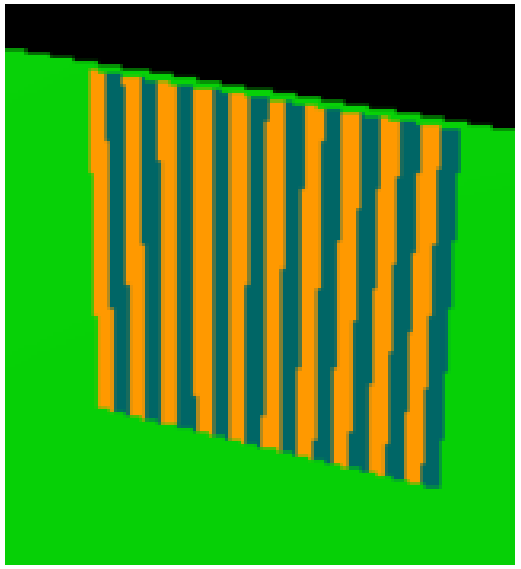
Texture Sampling





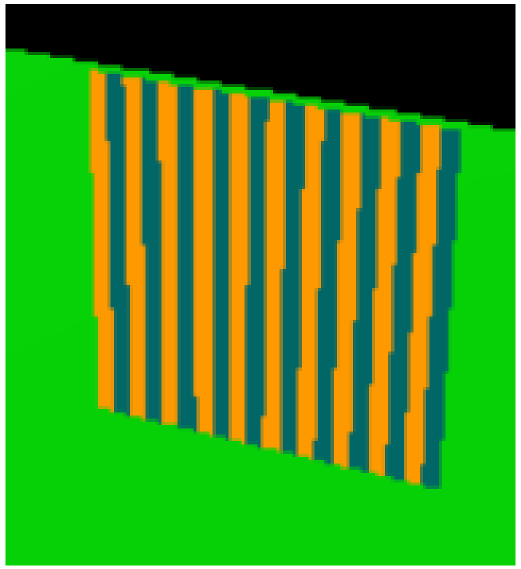
Jaggies are ugly

10 stripes

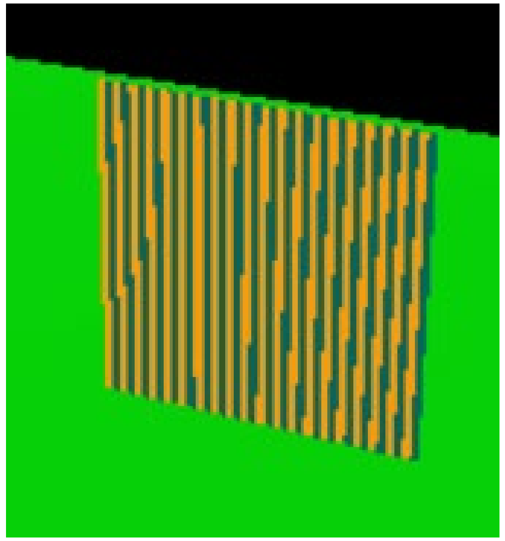


Jaggies are ugly and then get weird

10 stripes

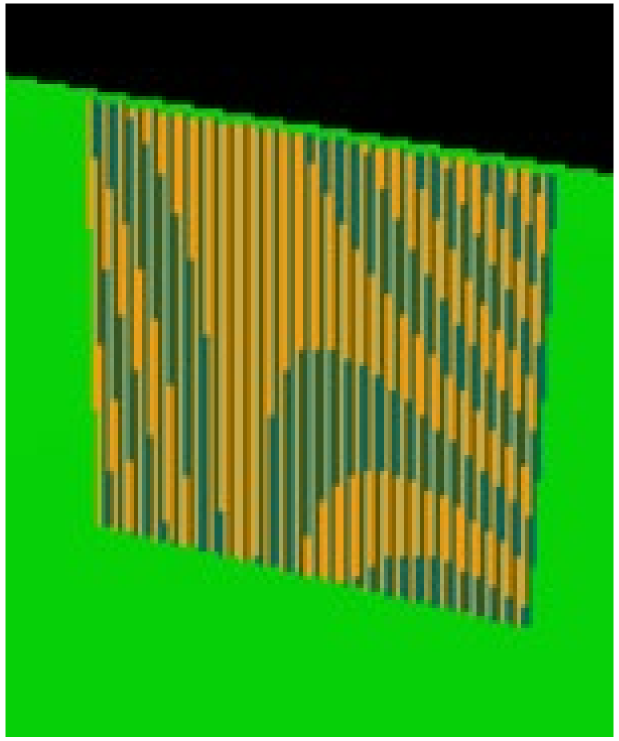


20 stripes

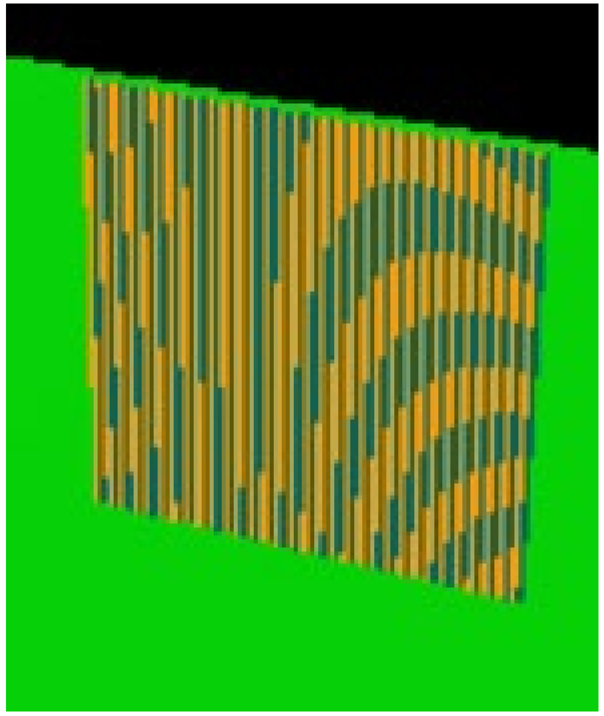


And really weird...

30 stripes

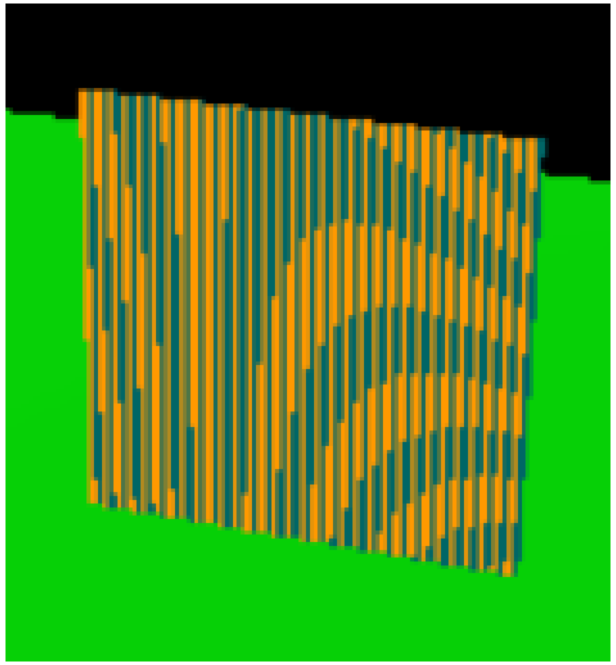


32 stripes

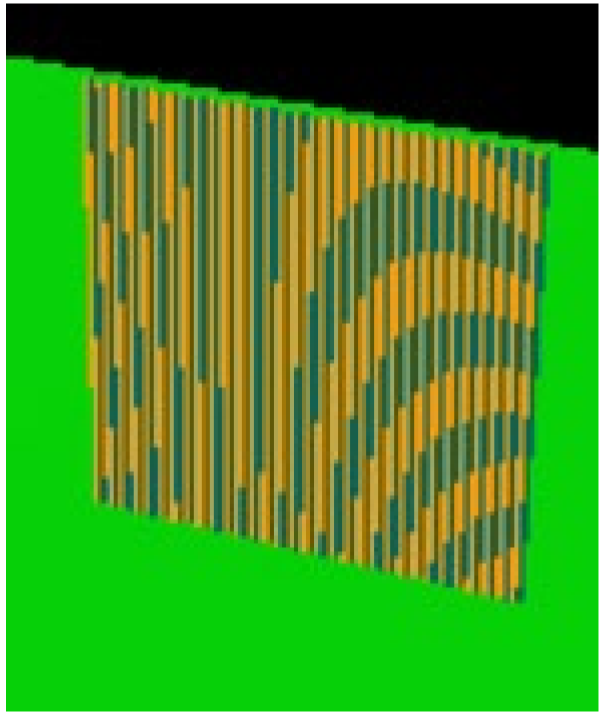


And really weird...

32 stripes (move the camera)



32 stripes



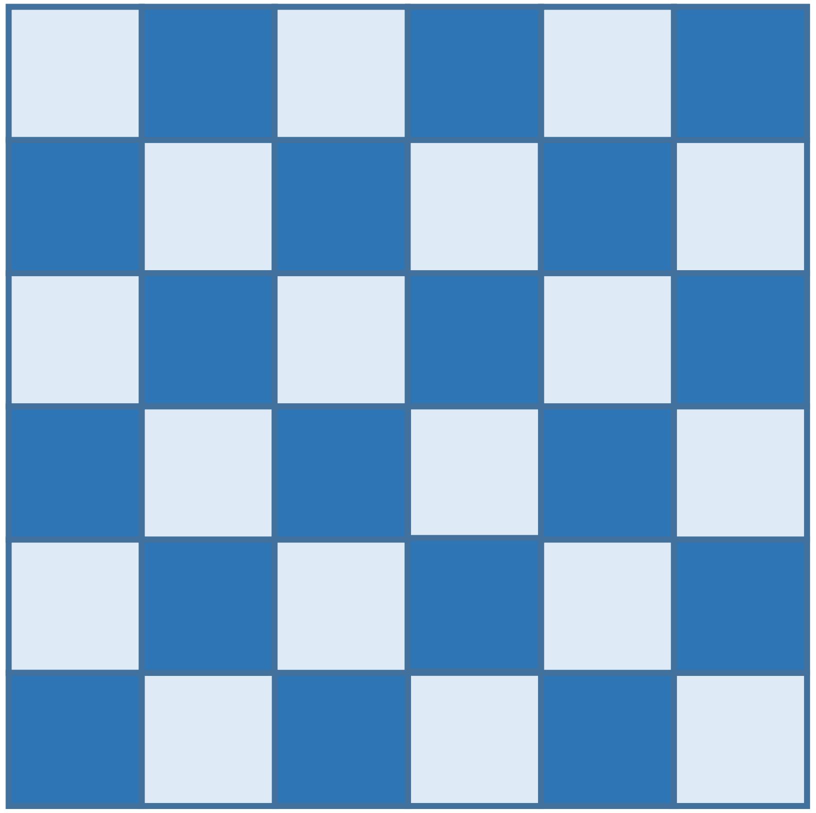
Even simple cases cause problems...

Make an image (e.g. texture) smaller...

Cut by a factor of 2

Even or odd

Gives very different results



The problem: Aliasing

We are making discrete choices

each pixel can be 1 color

Finite/discrete set of pixels

Infinite/continuous world

We are going to lose something

The solution?

More pixels!

Use the pixels we have better: **anti-aliasing**

Lecture 23 Part B
Anti-Aliasing Intuitions

Anti-Aliasing?

Warning: we can’t “fix” aliasing

We can lower our expectations

Avoid situations that we know will be problematic

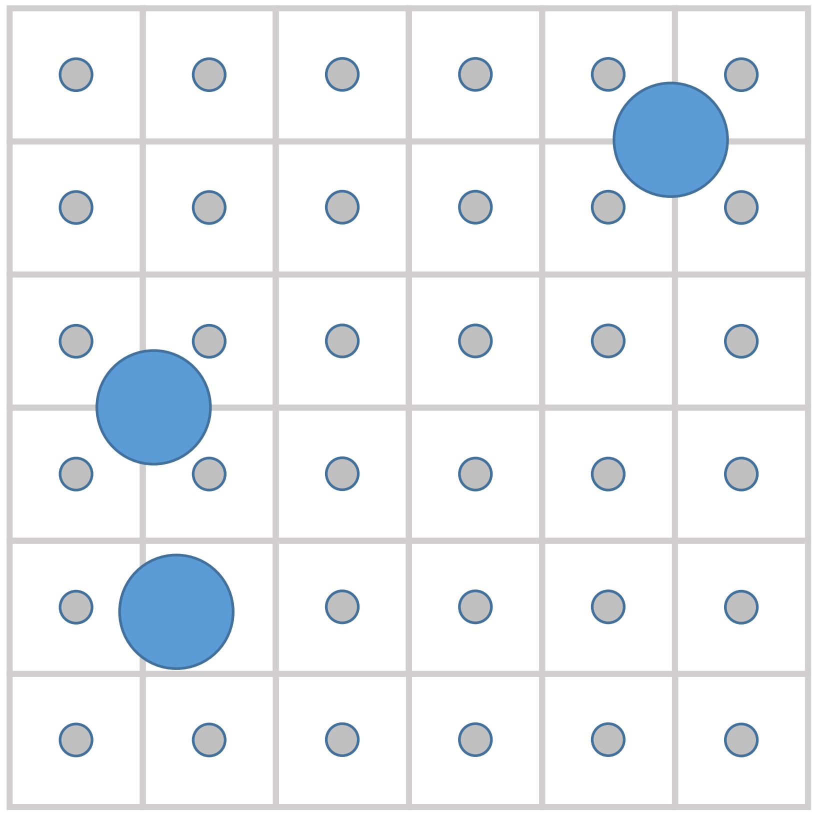
Anti-aliasing seeks to PREVENT bad aliasing

Over simplified version...

Over-Simplified Version...

Small dots are bad

might miss them
might blink as they move

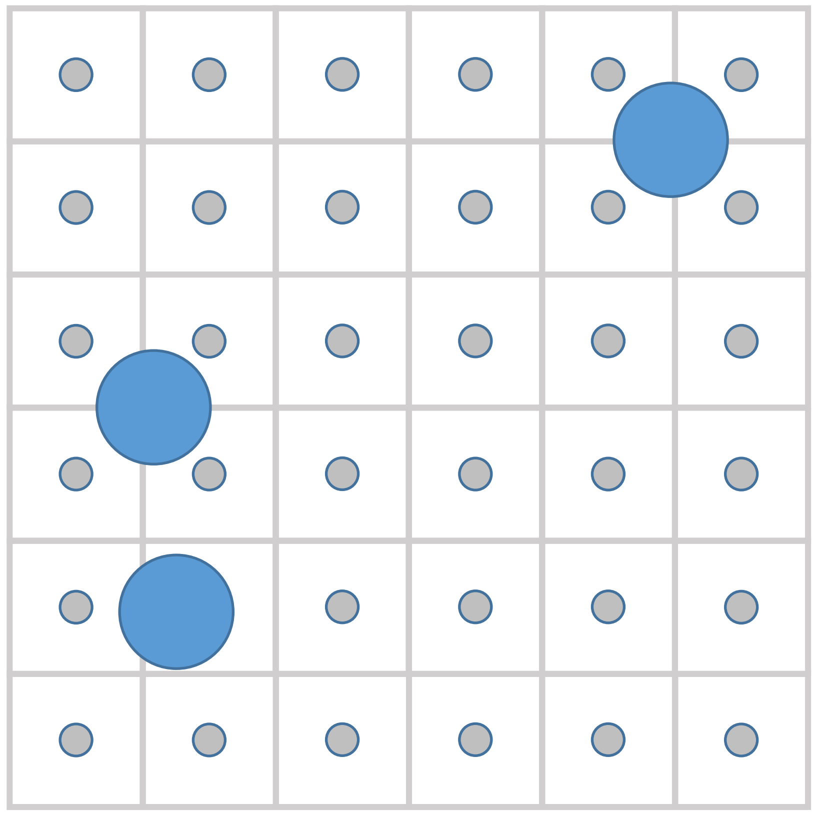


Over-Simplified Version (1)...

Small dots are bad

might miss them

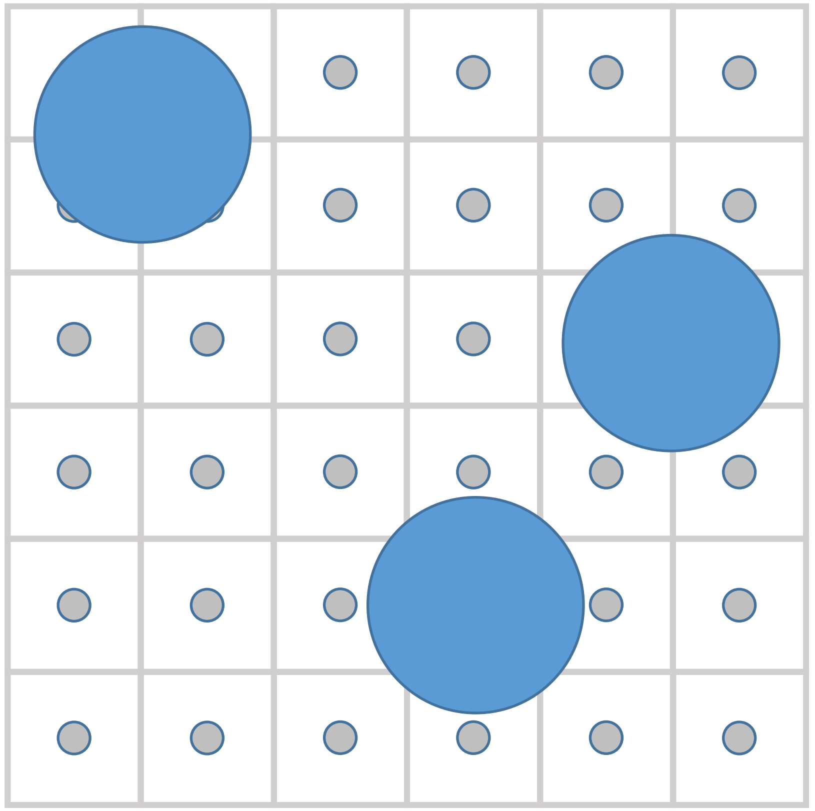
might blink as they move



Over-Simplified Version (2)...

Big dots are not a problem

We cannot miss them

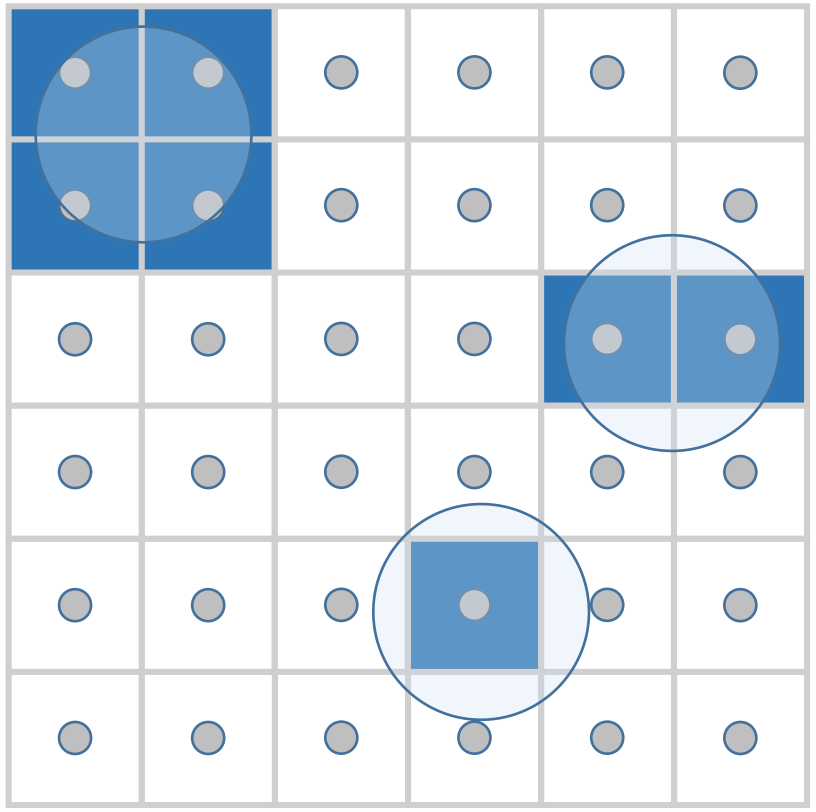


Over-Simplified Version (2b)...

They may change size and jump pixel to pixel

But we'll never lose them

(this is the simplified version)



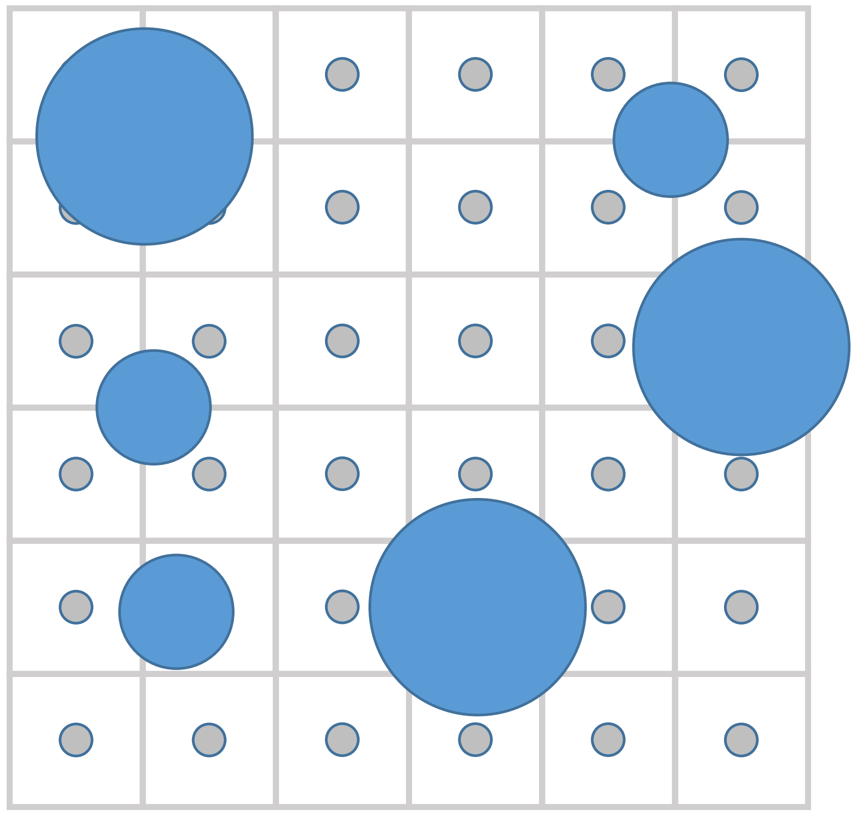
Over-Simplified Version (3)...

Small dots may be missed

can't count on them

Get rid of dots that are too small!

That way we'll be consistent



Filtering...

Get rid of things that might be a problem

Consistently get rid of things

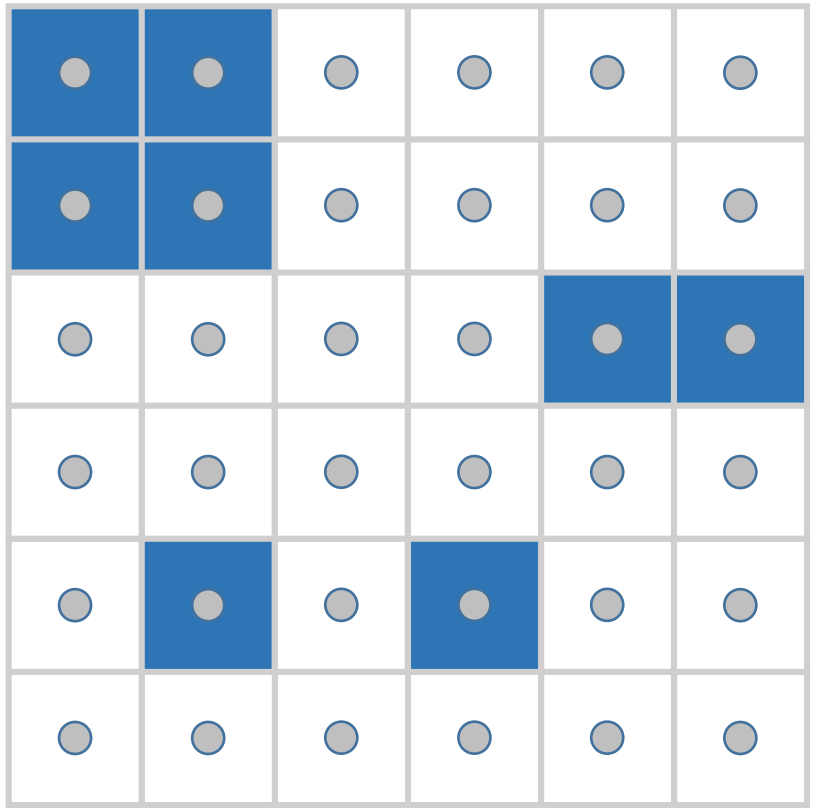
Get rid of potential problems BEFORE they happen

Over-Simplified Version (4)...

Once you've sampled, you've lost

You don't know what is a problem, and what isn't

Which one came from a dot that was too small?



Anti-Aliasing

Filter out potential problems before they happen

Pre-filtering

But it’s not about small dots…

Less Simplified Version

What matters is fast changes (sharp edges)

If things change quickly

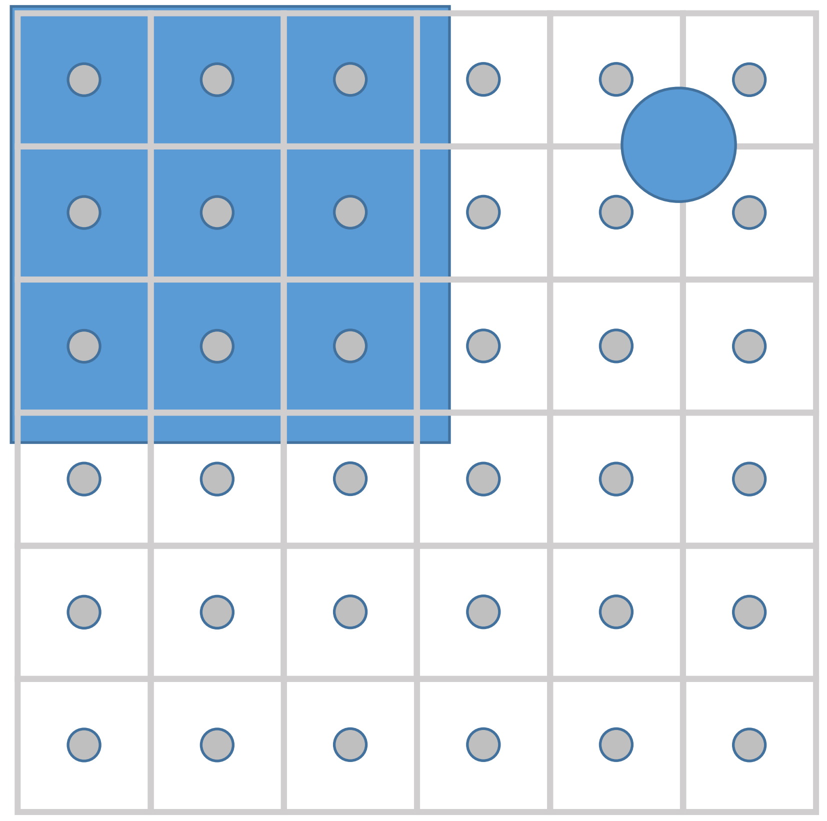
- small objects are possible
- sharp edges are hard to localize

In point sample land

Small things are bad

Sharp edges are bad
(edges are “small things”)

Worse when things move



Blur!

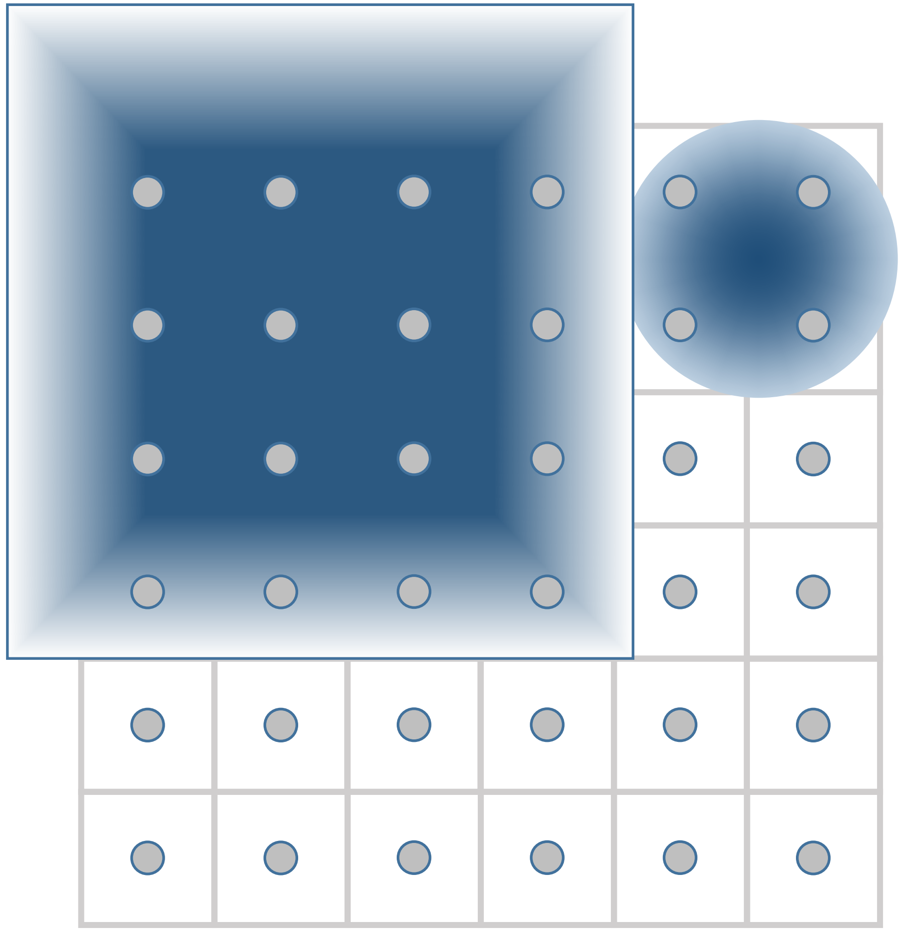
Small things are bad

Make them bigger!

Sharp edges are bad

Make them smooth!

Blurry is predictable!



Fast Changes Cause Problems

Actual Math...

Fast changes = high frequencies

Fourier transform (make things from sine waves)

fast changes need faster sine waves

No fast sine waves

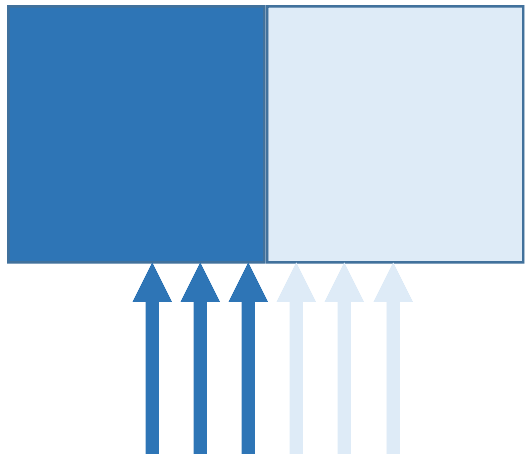
Low pass filter to get rid of high frequencies

Intuition: Sharp Edges are bad

Sharp Edge:

Small change in position

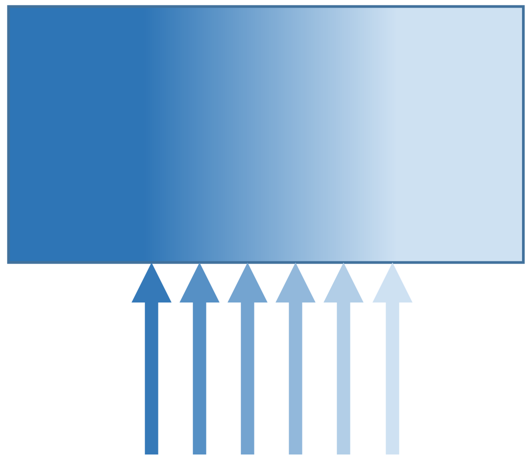
Big change in value



Smoother "Edge:"

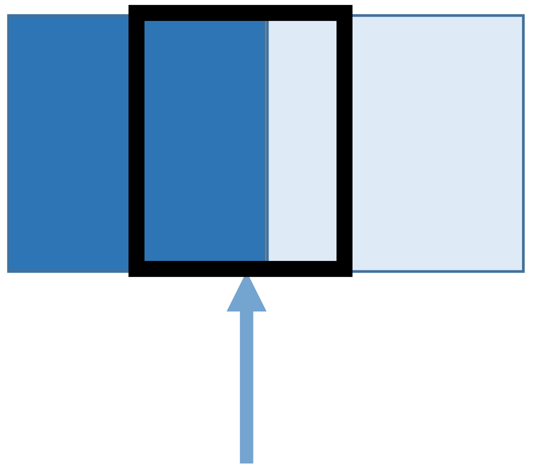
Small change in position

Doesn't matter (that much)



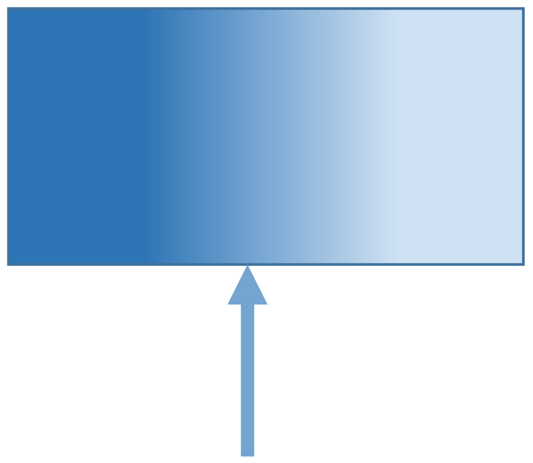
Idea: Average over the region

Average over the region

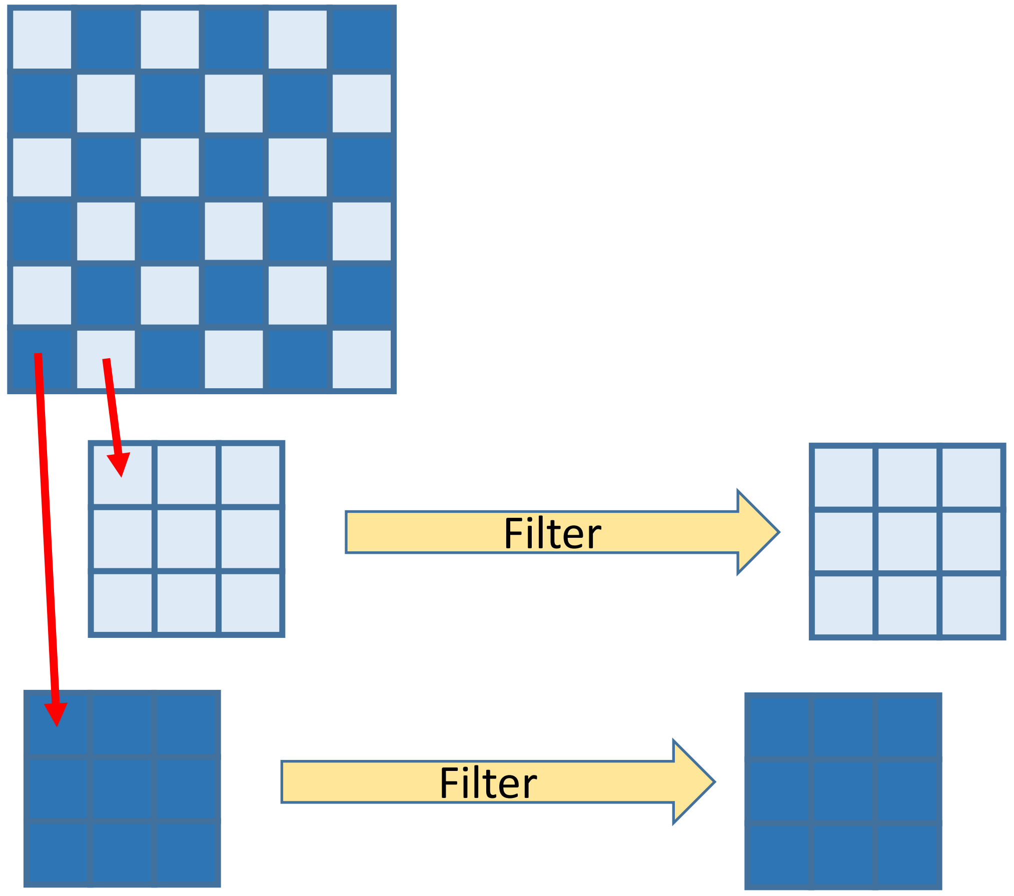


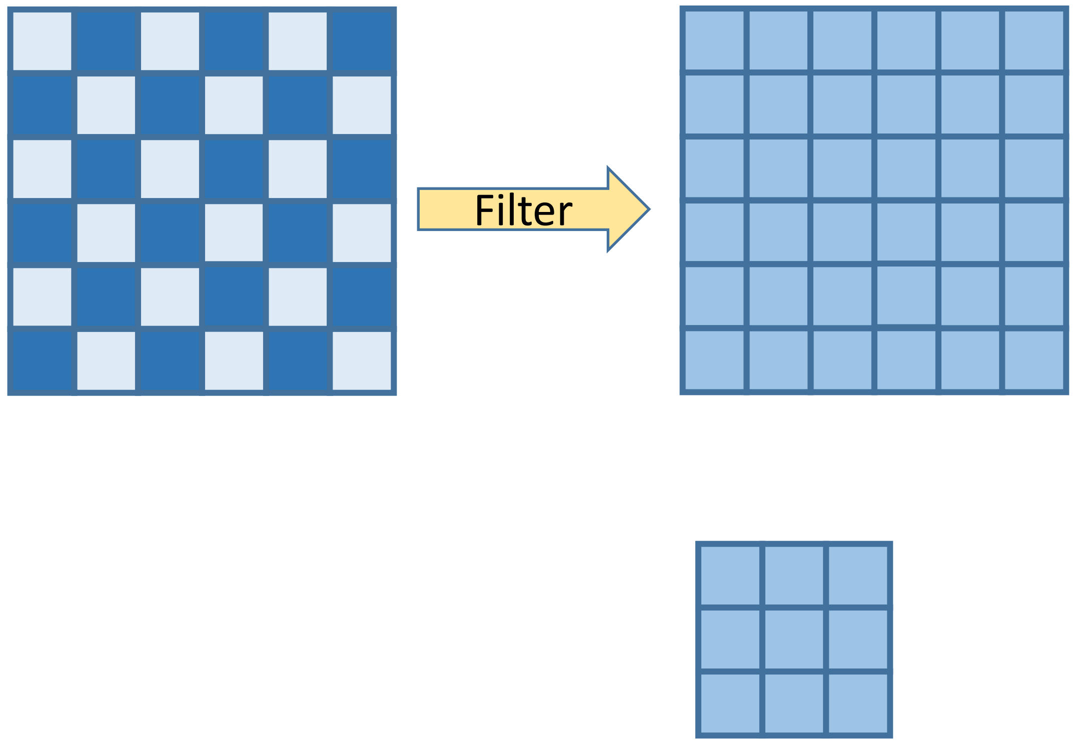
Pre-filter (blur)

Then point sample



Important: it is PRE-filtering you must filter before Aliasing occurs!





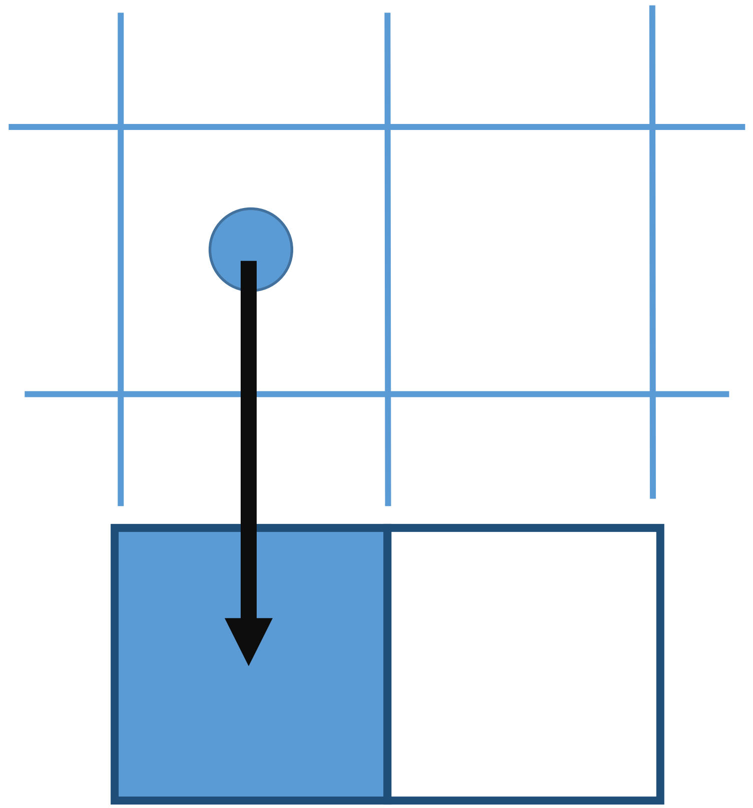
Anti-Aliasing

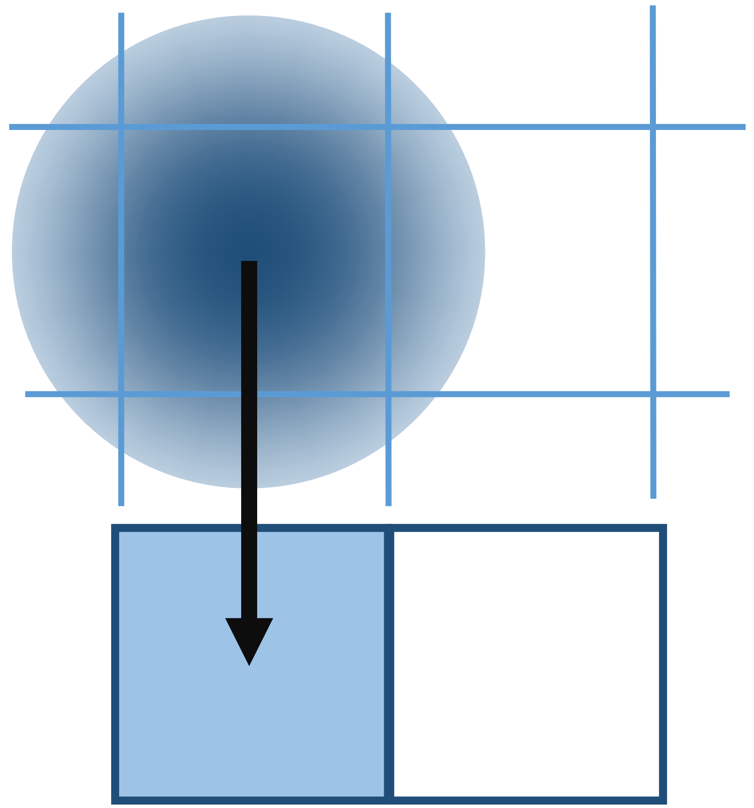
You cannot fix aliasing after it happens!

Take steps beforehand to avoid the worst problems

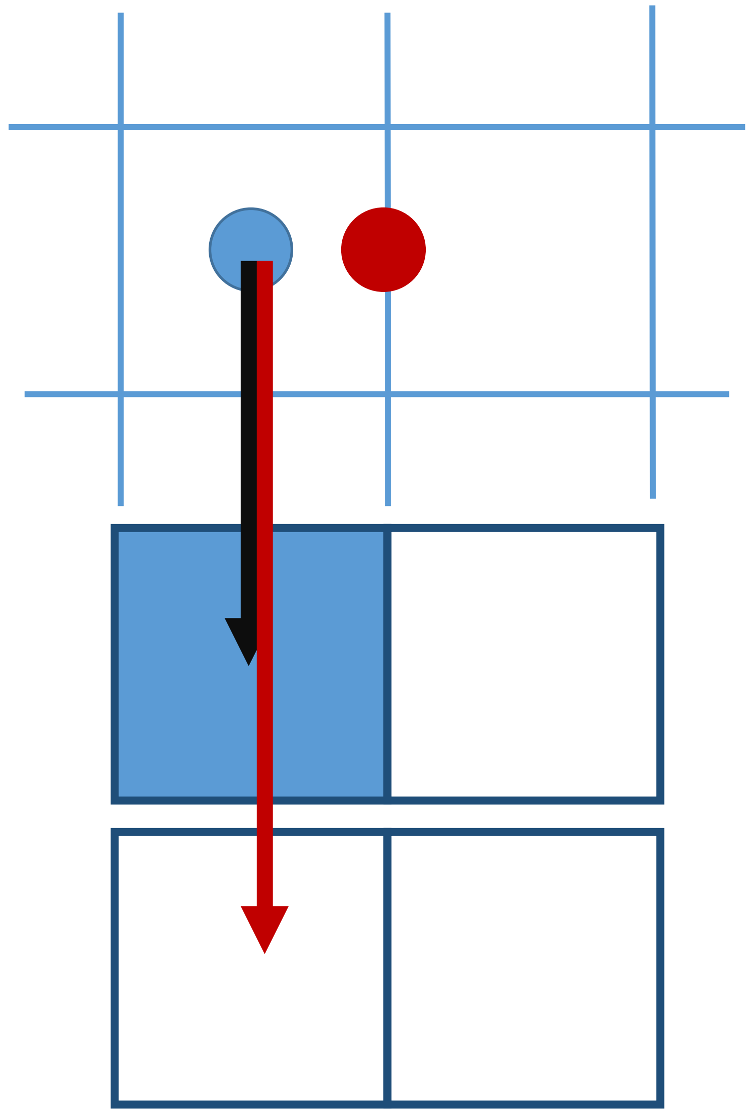
Blurry is better than wrong

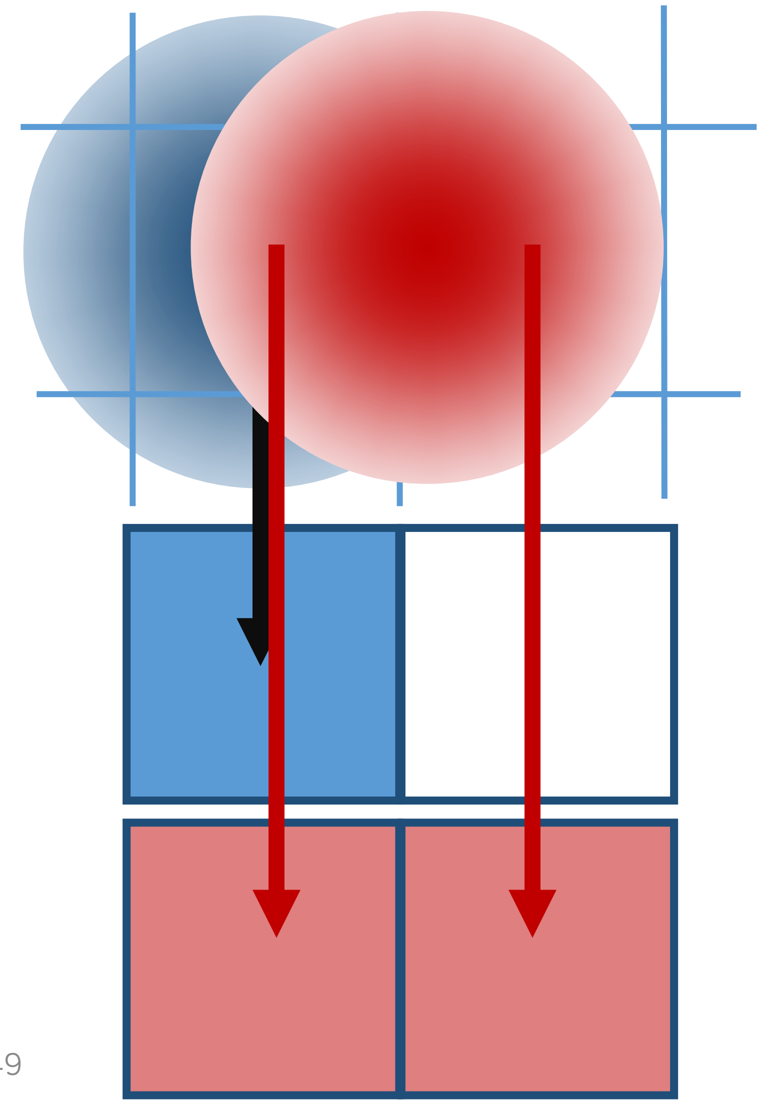
How does this help the point problem?



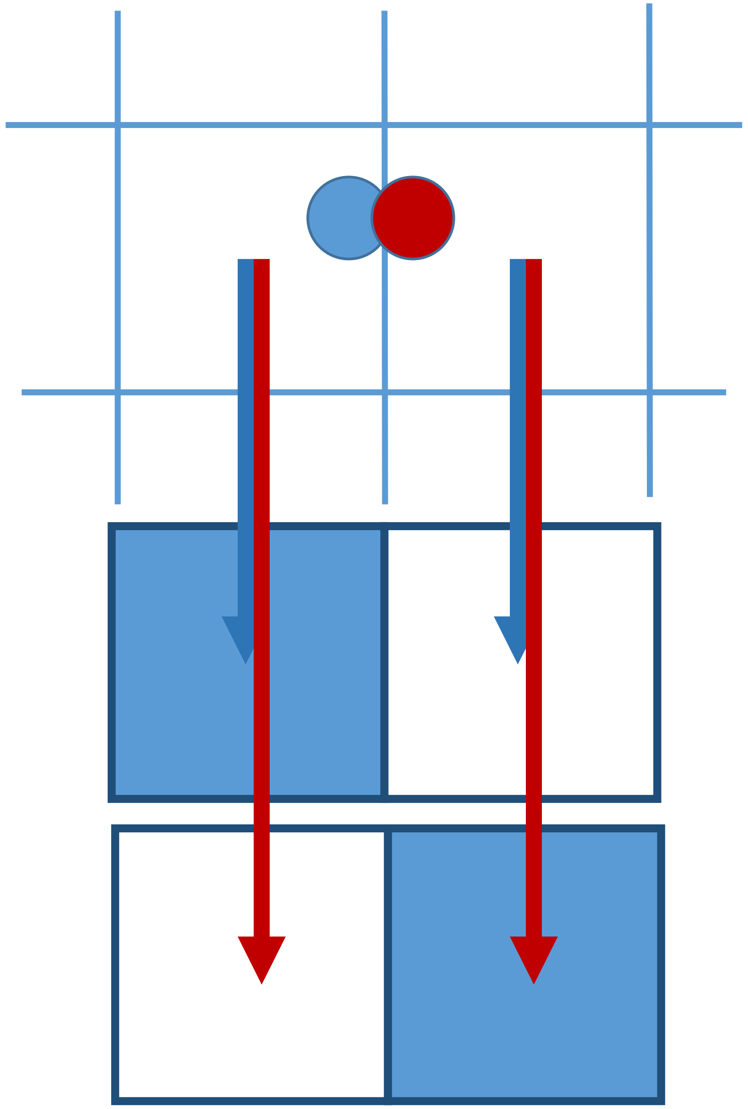


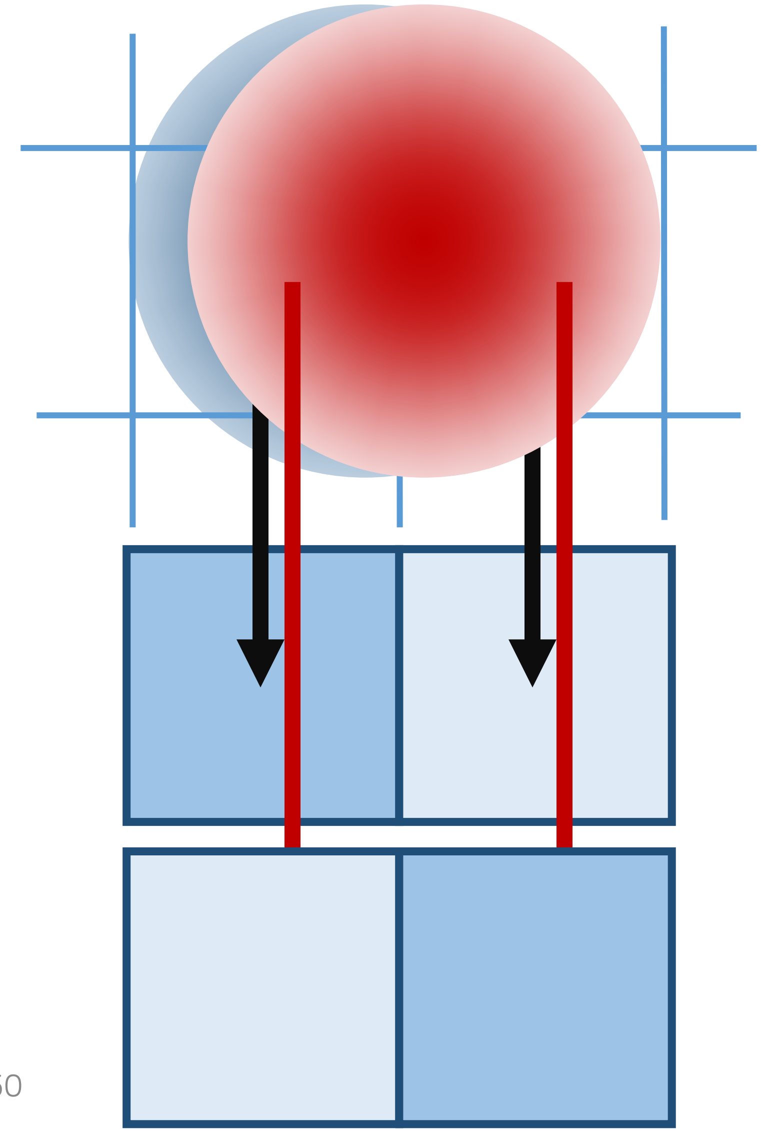
How does this help the point problem?





How does this help the point problem?

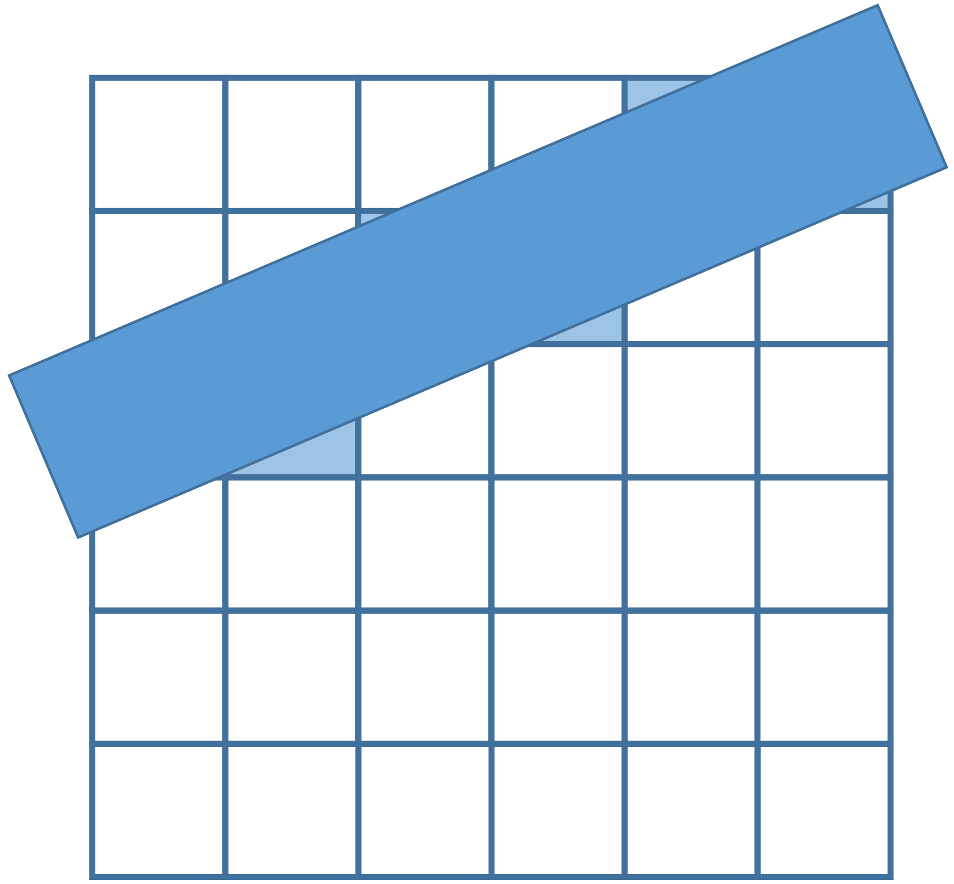




Lines

Make line tick enough?

Still have “fast changes”



Two ways to view it...

Thick line

Blur

Measure at point

(distance point to line)

Thick line

Area Coverage

Anti-Aliasing Primitives

It can’t be a binary yes/no decision

Primitive can partially fill a pixel

Blurred primitive can partially cover the sample point

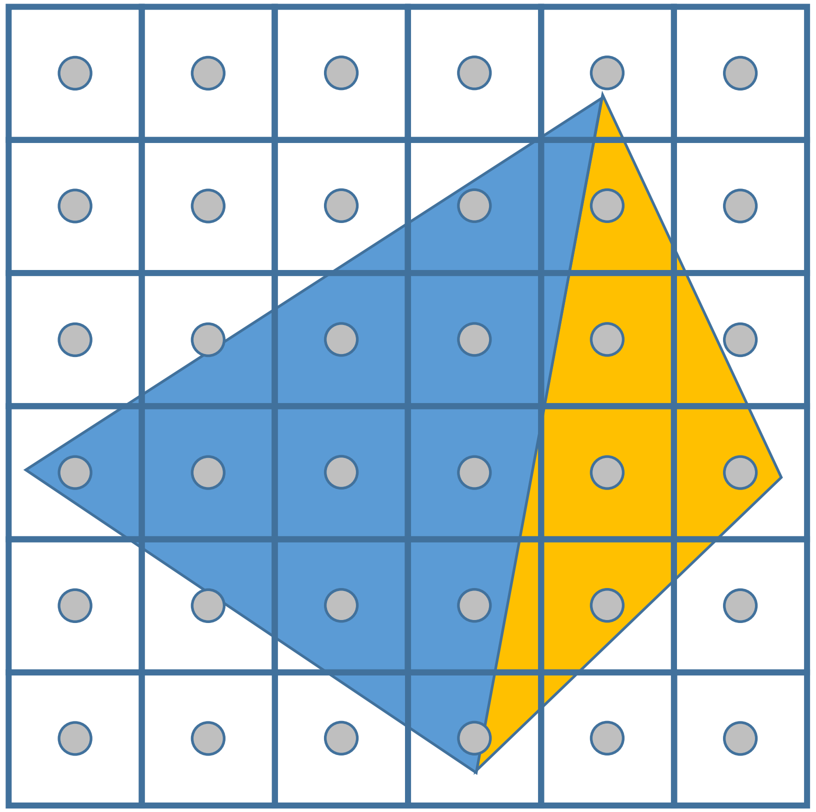
Problem: what fills the other part?

Partial Fill and Triangles

A pixel may involve two (or more) triangles!

Easy to understand with "little square square coverage"

Also edge distance to center point, center point measures blurred edge, etc.



Anti-Aliasing Triangle Edges

Have pixels keep track of multiple triangles?

Hard (lots to store per pixel)

We want to keep triangles independent

Keeping Primitives Independent

Partial fill against background

Alpha channel (transparency)

Drawing order matters

This is how 2D drawing works (high quality)

In Practice…

Anti-aliasing triangle edges is problematic

Use transparency (partial filling) when possible

requires back to front (OK for 2D)

Problem is worse when we consider visibility

Z-Buffer is a form of aliasing (yes no per pixel)

Within a triangle… much easier

Textures are easier to filter – use big triangles with texture