Ray Tracing Notes

## 4.2

At its core, a ray tracer sends rays through pixels and computes the color seen in the direction of those rays. The involved steps are

1. Calculate the ray from the “eye” through the pixel,
2. Determine which objects the ray intersects, and
3. Compute a color for the closest intersection point.

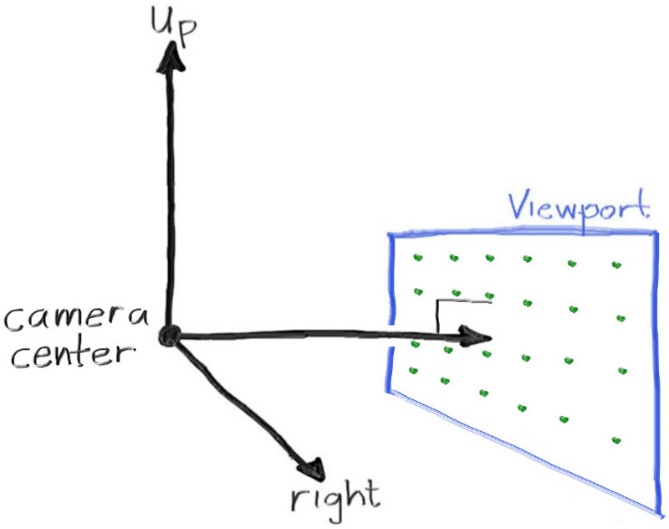
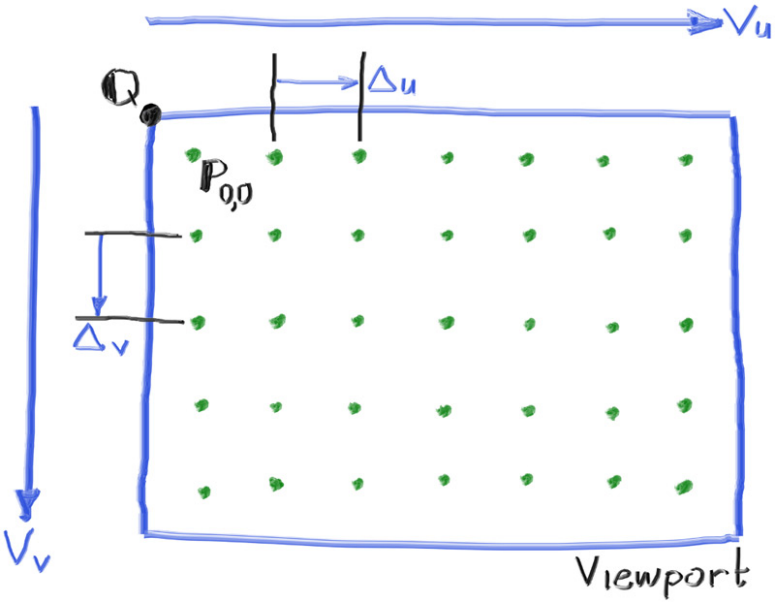
Given an image with a 16∶9 aspect ratio, width/height=16/9=1.7778

* A **viewport** is the rectangular region of the window where rendering happens. A **virtual viewport** means you render as if the screen has a certain resolution, then scale that rendering to fit the real window. This is useful for keeping consistent aspect ratio and UI scaling predictably across devices. Example: A game might render to a virtual viewport of 1920×1080, then scale that to a phone screen of 1170×2532.
* **Pixel spacing** – distance between two pixels

We want to make sure that the image height is never negative or below 1 when rendering the image on a viewport by using a safety clamp: image\_height = (image\_height < 1) ? 1 : image\_height

**? 1 : image\_height** is a ternary operator that chooses between two values.

If True, use 1. If False, use image\_height

* Imagine rays shooting from the camera lens (camera center) to all the pixels (“many diagonal lines”)

A **lerp** is a standard equation defined by lerp(a, b, t) = a \* (1 - t) + b \* t

