



*COMP3005: Computer Vision*

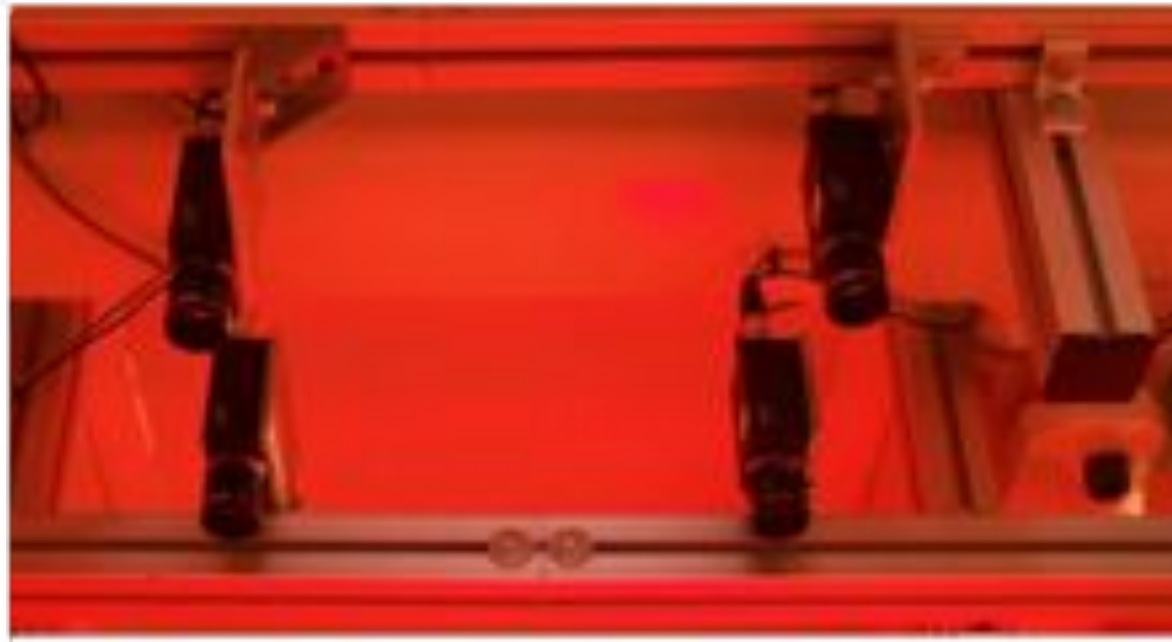
---

# Building machines that see

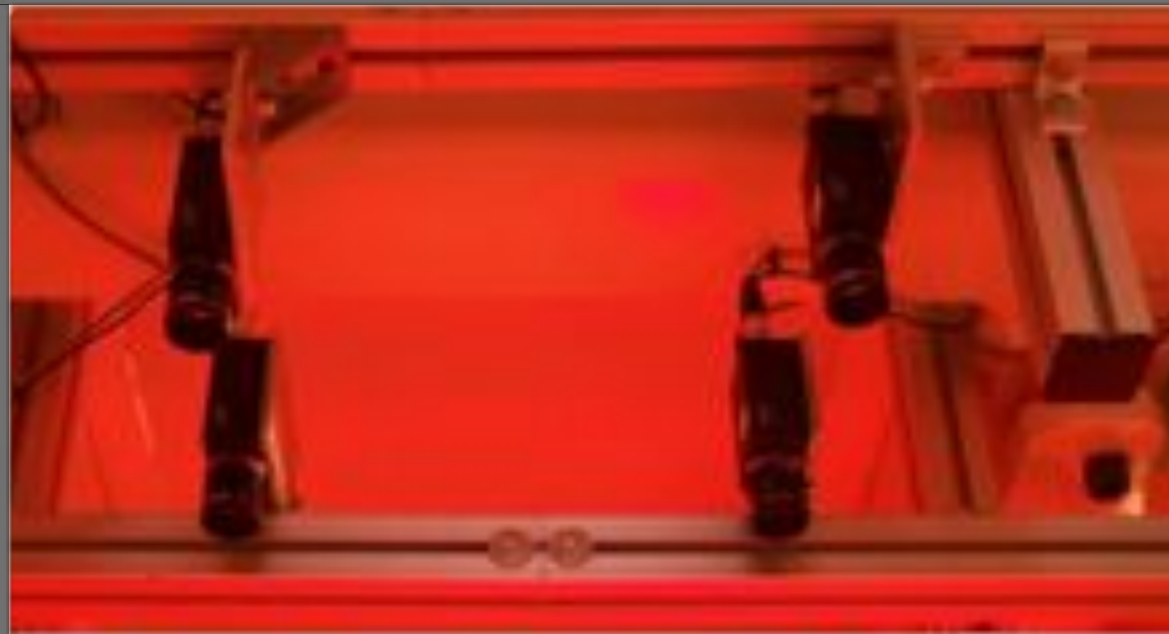
Jonathon Hare  
[jsh2@ecs.soton.ac.uk](mailto:jsh2@ecs.soton.ac.uk)

---

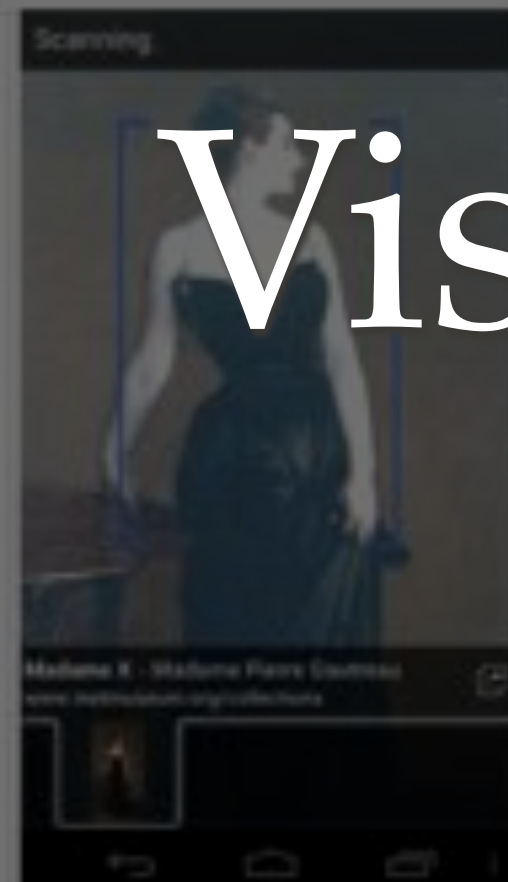
# Types of Computer Vision and their **Environment**



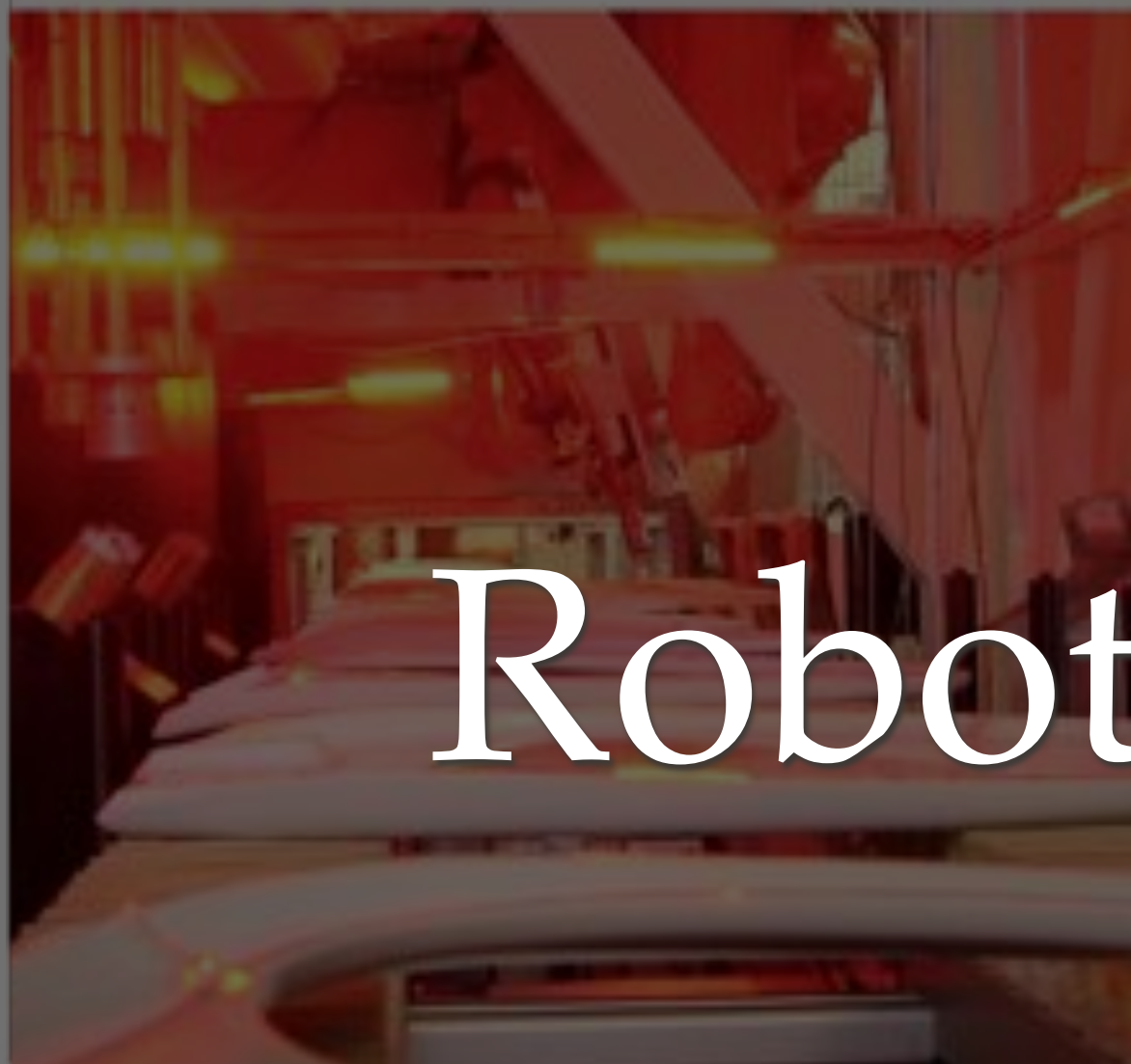




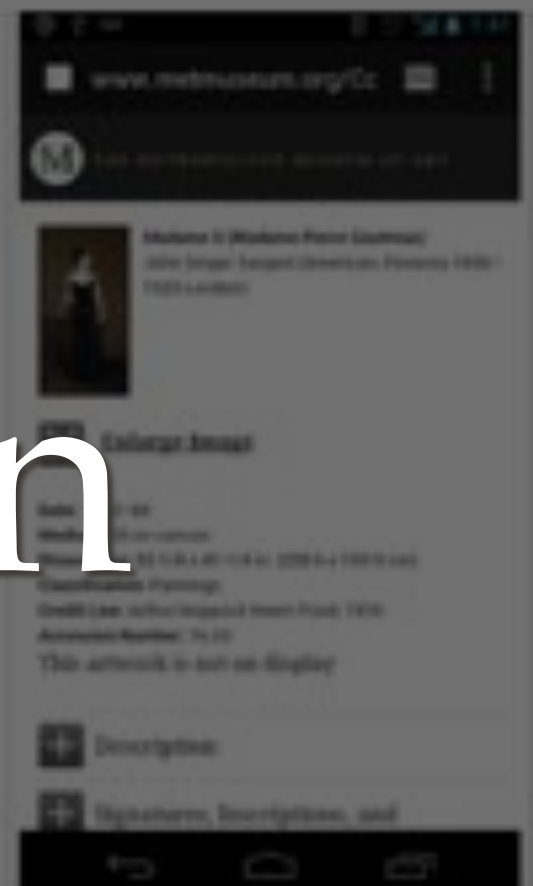
# Industrial Vision







# Robot Vision



# Vision in the wild





What do all these systems have in  
common?

# Computer Vision Software





# Image Acquisition Hardware



but how do you go about designing  
a computer vision system? and is  
that all you need?



---

# Key terms in designing CV systems

---

**robust**

**invariant**

**repeatable**

**constraints**

---

# Key terms in designing CV systems

---

**robust**

**invariant**

**repeatable**

*These are what you want*

**constraints**



---

# Key terms in designing CV systems

---

**robust**

**invariant**

**repeatable**

*This is what you design  
your system to be*

---

# Key terms in designing CV systems

---

**robust**

**i** *This is what you apply* **e**  
*to make it work*

**constraints**



---

# Robustness

---

- ❖ The vision system must be **robust** to changes in its environment
- ❖ i.e. changes in lighting; angle or position of the camera; etc

---

# Repeatability

---

- ❖ **Repeatability** is a *measure* of robustness
- ❖ Repeatability means that the system must work the same over and over, regardless of environmental changes

---

# Invariance

---

- ❖ **Invariance** to environmental factors helps achieve robustness and repeatability
- ❖ Hardware and software can be designed to be invariant to certain environmental changes
  - ❖ e.g. you could design an algorithm to be invariant to illumination changes...

---

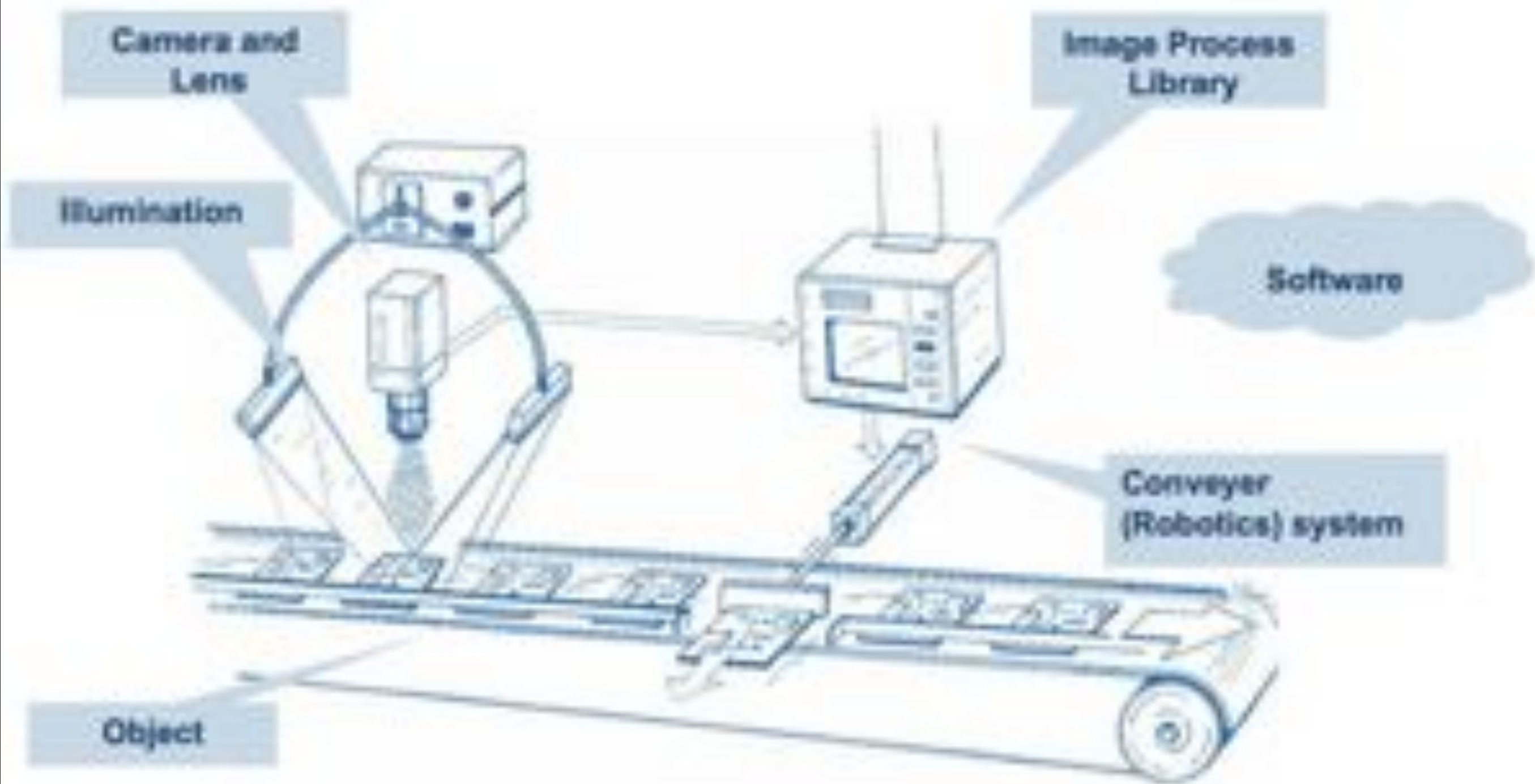
# Constraints

---

- ❖ **Constraints** are what you apply to the hardware, software and wetware to make your computer vision system work in a repeatable, robust fashion.
- ❖ e.g. you constrain the system by putting it in a box so there can't be any illumination changes



# Constraints in Industrial Vision



---

# Software Constraints

---

- ❖ Really simple, but incredibly fast algorithms
- ❖ Hough Transform is popular, but note that it isn't all that robust without physical constraints
  - ❖ Actually, same is true of most algorithms / techniques used in industrial vision
- ❖ Intelligent use of colour...

---

# *Important aside:* Colour-spaces

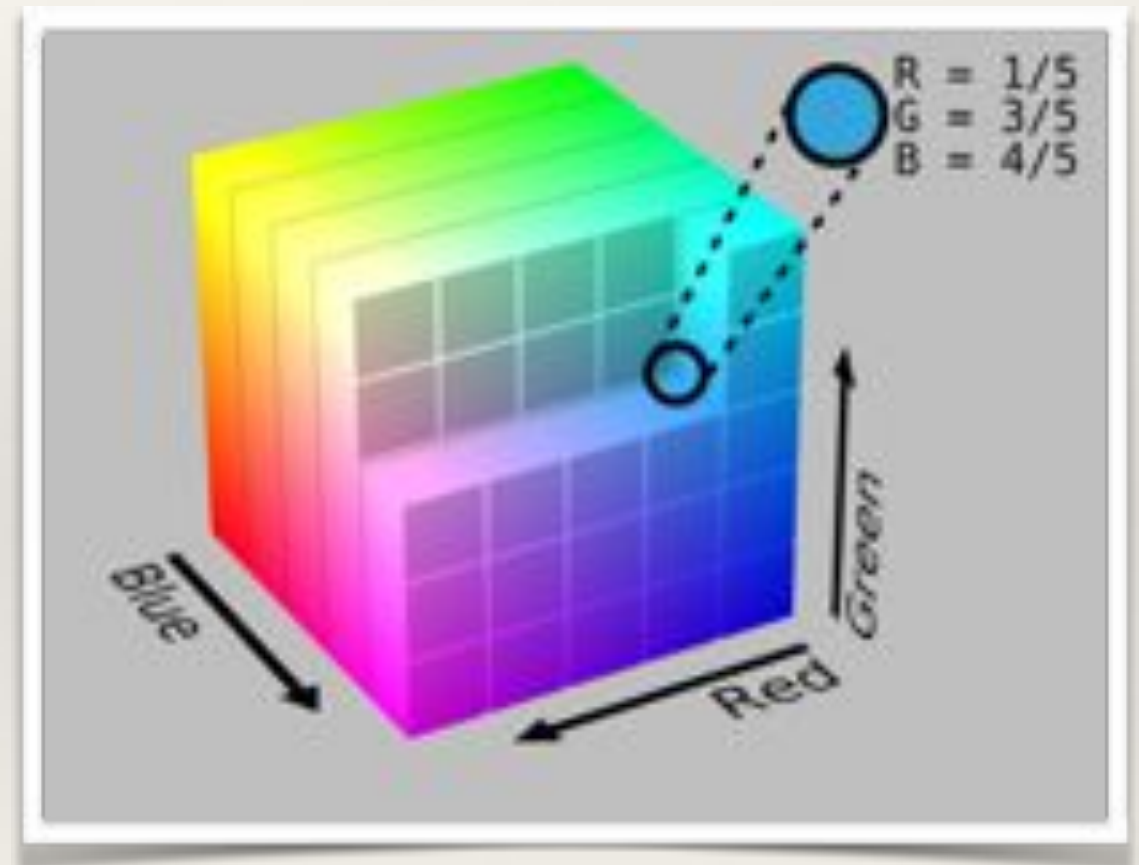
---

- ❖ There are many different ways of *numerically* representing colour
- ❖ A single representation of all possible colours is called a colour-space
- ❖ It's *generally* possible to convert to one colour-space to another by applying a mapping (in the form of a set of equations or an algorithm)



# RGB Colour-space

- ❖ Most physical image sensors capture RGB
- ❖ By far the most widely known space
- ❖ RGB “couples” brightness (luminance) with each channel, meaning that illumination invariance is difficult.



# HSV Colour-space

- ❖ Hue, Saturation, Value is another colour-space
  - ❖ Hue encodes the pure colour as an angle
    - ❖ **red == 0° == 360° !!**
  - ❖ Saturation is how vibrant the colour is
  - ❖ And the Value encodes brightness
  - ❖ A simple way of achieving invariance to lighting is to use just the H or H & S components



# *Demo: colour-spaces*

---

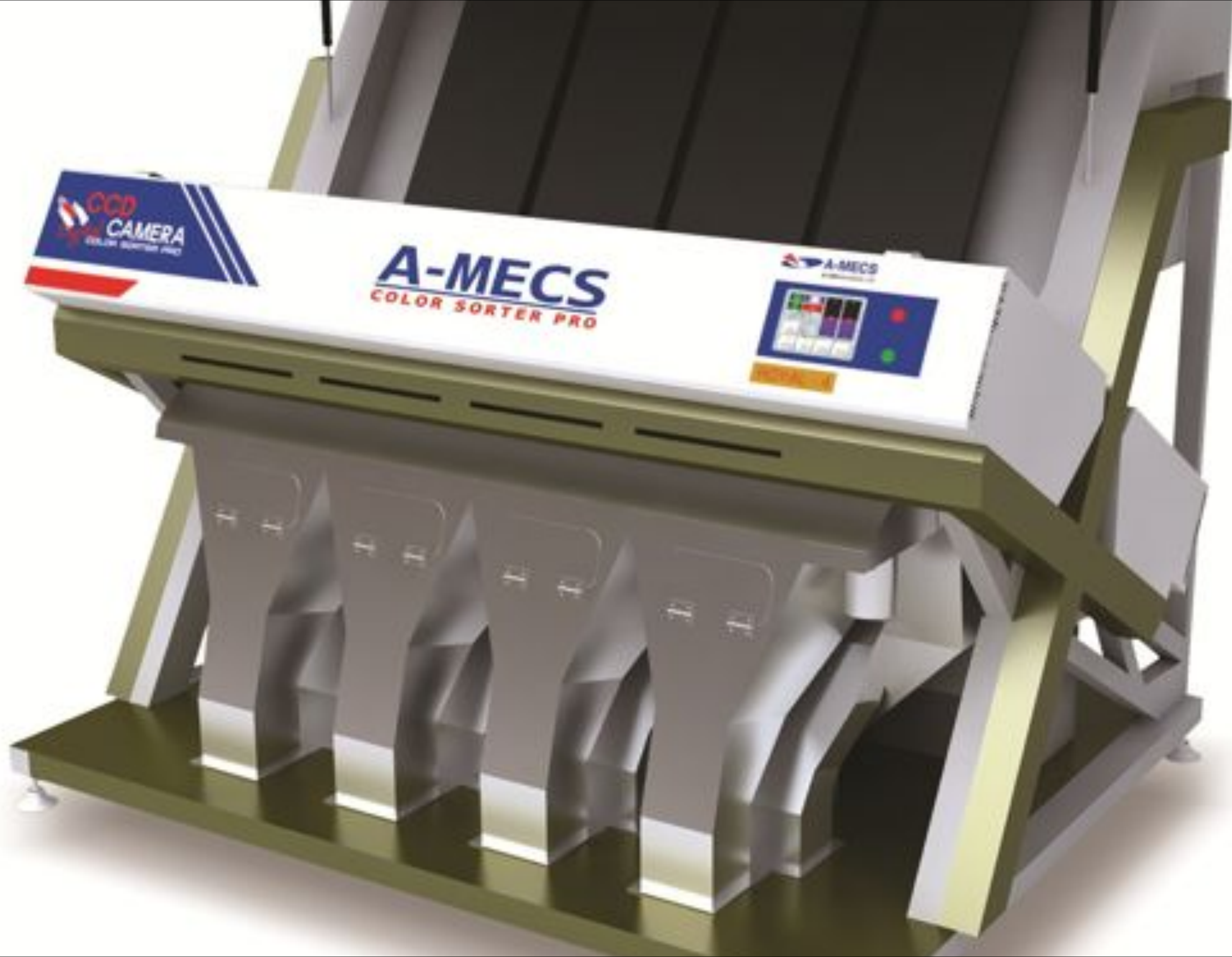
# Physical Constraints

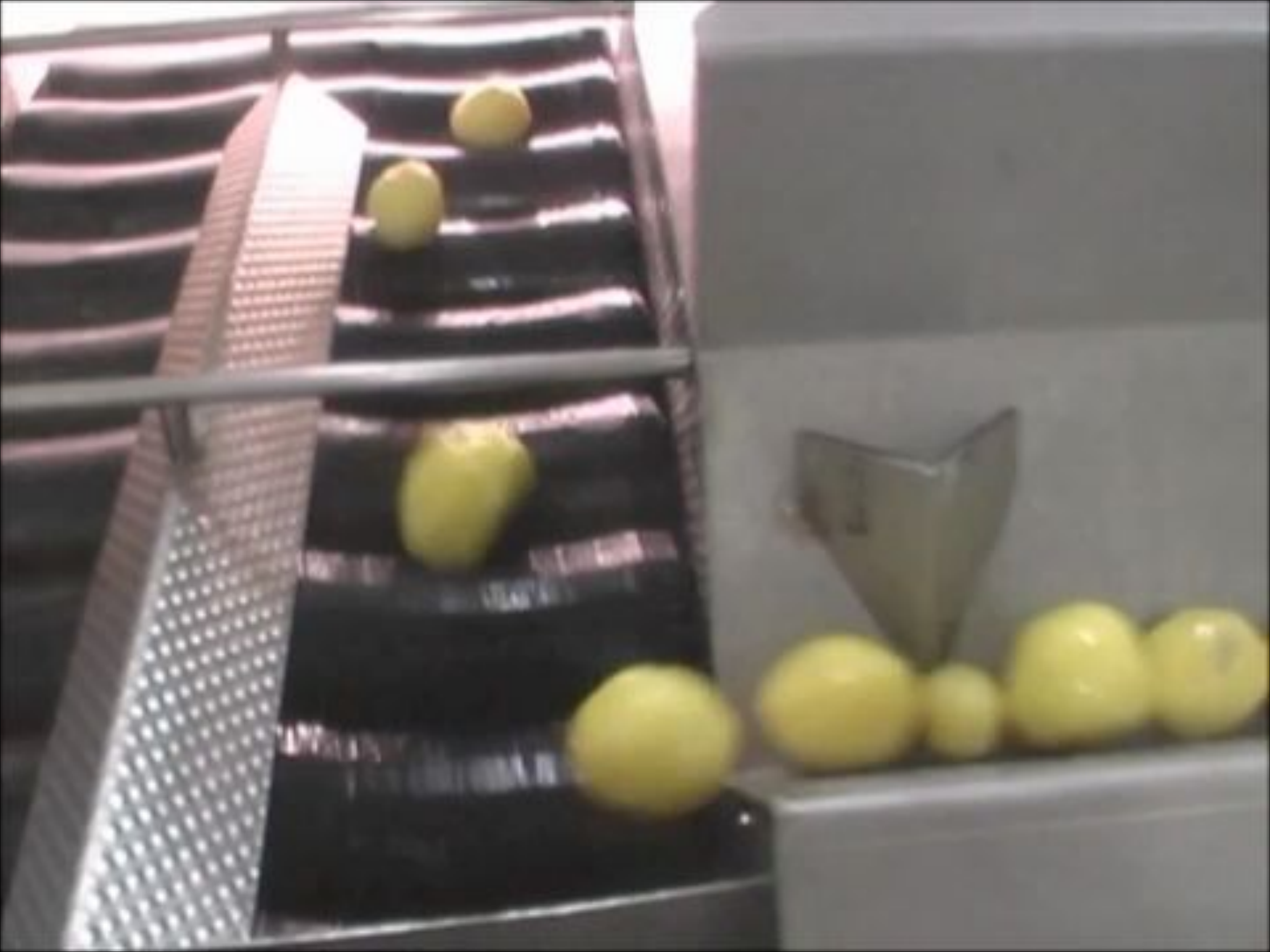
---

- ❖ Industrial vision is usually solved by applying simple computer vision algorithms, and lots of physical constraints:
- ❖ Environment: lighting, enclosure, mounting
- ❖ Acquisition hardware: expensive camera, optics, filters



*Let's look at some types of physical  
constraint*





---

# Vision in the wild

---

- ❖ So, what about vision systems in the wild, like ANPR cameras, or recognition apps for mobile phones?
- ❖ Apply as many hardware and wetware constraints as possible, and let the software take up the slack
- ❖ Colour information often less important than luminance



# ANPR constraints

- ❖ License plate styles are different across the world, so most ANPR systems will only work with plates from a single country.
- ❖ License plates themselves are constrained in design:
  - ❖ Dimensions
  - ❖ Font
  - ❖ Material (IR reflectance!)



# Mobile vision constraints

- ❖ QR-Codes are designed to be robust
- ❖ But most software requires (constrains) the user to operate in a certain way:
  - ❖ Orientation - approximately upright
  - ❖ Within a certain area
  - ❖ Approximately stationary



---

# Almost unconstrained vision?

---

- ❖ As computers become more powerful, and new software techniques are developed to deal with invariance the need for constraints becomes less.
- ❖ ...but there is always going to be a problem of optimising the costs, and constraints can always help reduce costs