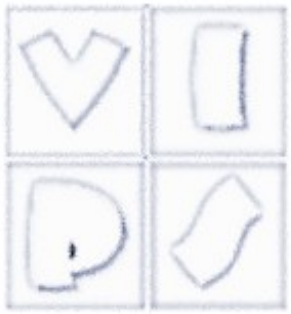
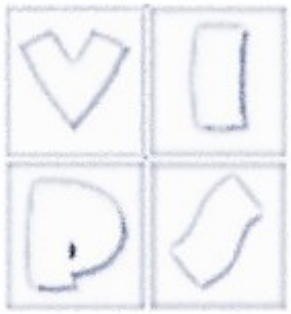


Fondamenti di Grafica al calcolatore

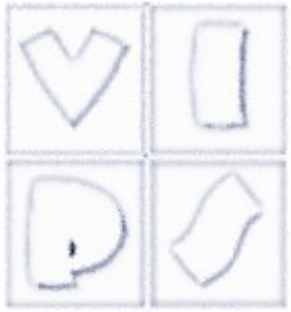
Andrea Giachetti
Department of Computer Science, University of Verona, Italy

andrea.giachetti@univr.it



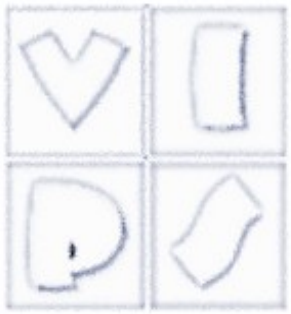
Il corso

- Fondamenti di computer grafica, da definire anche in rapporto alle vostre competenze di base
- Argomenti indicativi
  - Applicazioni grafiche, immagini raster, colore, display e dispositivi di input
  - Geometria dello spazio e modellazione
  - Teoria del rendering (interazione luce/materia)
- Modalità di esame:
  - Prova scritta
- Docente: Andrea Giachetti
  - Ricevimento
    - Stanza 1.86



Materiale didattico

- Sufficienti: lucidi del corso, messi a disposizione su
  - www.andreagiachetti.it
- Altri libri per riferimenti/approfondimenti
  - E. Angel, Interactive Computer Graphics with OpenGL, OpenGL, 6th edition, Addison Wesley 2011
  - R. Scateni, P. Cignoni, C. Montani, R. Scopigno, Fondamenti di graca tridimensionale interattiva, McGraw-Hill, 2005
  - F. Ganovelli, M. Corsini, S. Pattanaik, M. Di Benedetto, Introduction to Computer Graphics: A Practical Learning Approach, Eds. CRC Press
  - S.R. Buss, 3D Computer Graphics, Cambridge University Press, 2003
  - S.M. LaValle Virtual Reality Cambridge University Press 2017
    - Gratis al sito http://vr.cs.uiuc.edu/



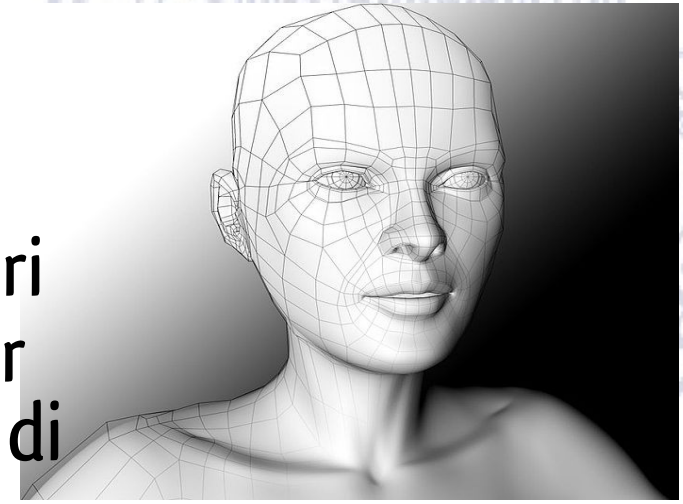
Il piano del corso

- Oggi
  - Introduzione, immagini digitali, colore, applicazioni grafiche
    - 3ore? Fine 11.30 poi alle 1230 siete invitati caldamente a seminario aula I
    - From 3D models to 3D prints: an overview of the processing pipeline
    - Timetable: h 12:30, Aula I | Speaker: Marco Livesu (IMATI CNR Genova)
    - Poi recuperiamo 1 ora quando volete
- Lezione 2
  - Geometria dello spazio e modellazione 30/11
- Lezione 3
  - Rendering 11/1
- Lezione 4
  - Rendering 1/2

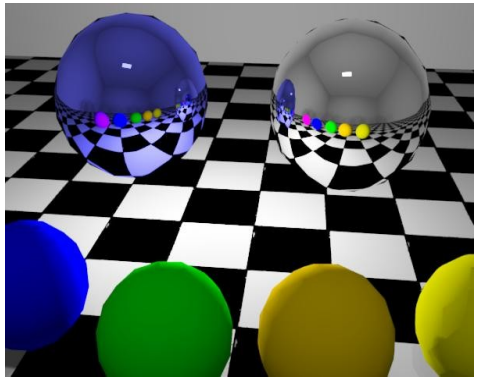
Grafica e videogames

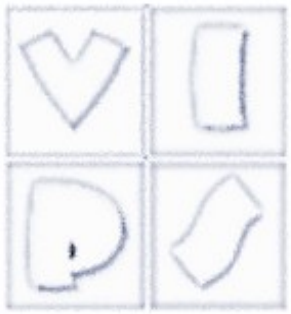
- Lo sviluppo delle due discipline è sicuramente largamente intersecato
- La comunità della Computer Grafica interattiva ha sviluppato i vari strumenti su cui si basano i moderni giochi
- E si occupa non solo del rendering visuale, ma anche di molti aspetti legati al gaming
  - Modellazione
  - Simulazione fisica
  - Animazione
  - Realtà virtuale in generale
- Qui ci occuperemo più che altro degli aspetti di base del rendering interattivo, ma ovviamente ci sarebbe molto di più, che però vedrete in altri corsi

Grafica al calcolatore



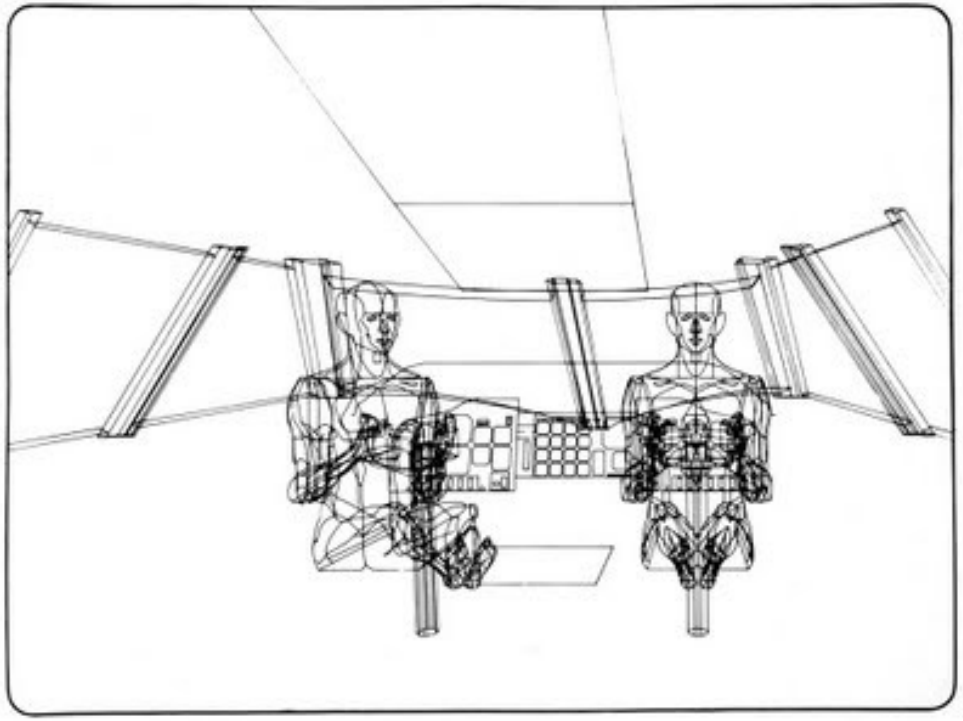
- Che cos'è?
  - Risposta meno ovvia di quanto sembri
- Intuitivamente: uso di un calcolatore per produrre un'immagine (o una sequenza di immagini)
  - Non necessariamente realistica o 3D
  - Non necessariamente interattiva
  - Ma le cose sono più complicate
- Per capire meglio vediamo un po' di storia
  - Nasce con i primi display per i calcolatori





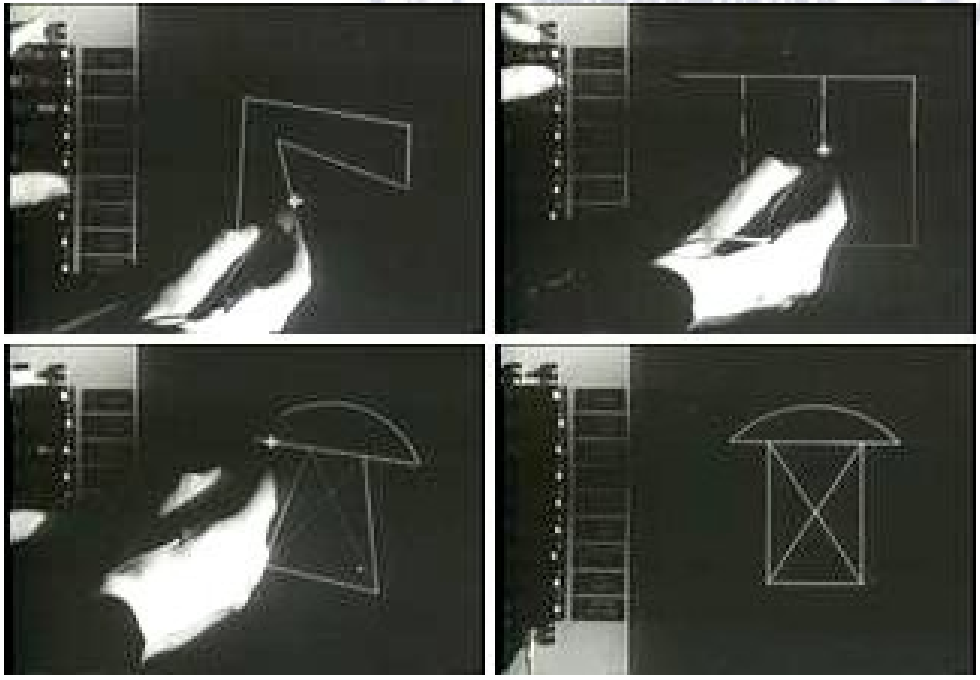
Storia

- 1960
  - William Fetter introduce il termine Computer Graphics per descrivere la ricerca che stava conducendo alla Boeing. Modello 3D del corpo umano per progettare la carlinga degli aerei.
  - C'è quindi l'idea della modellazione 3D
  - E' una parte rilevante della moderna CG



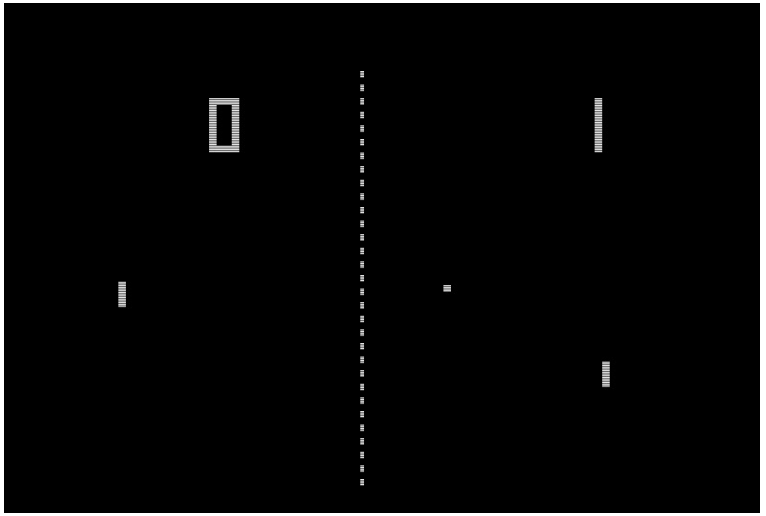
Storia

- 1963
  - Nascita della Computer Grafica interattiva: sistema sketchpad di Ivan Sutherland
  - In questo caso si tratta della prima **interfaccia grafica** interattiva
- Negli anni sessanta nascono i primi terminali grafici e giochi, si impara a disegnare sullo schermo 2D



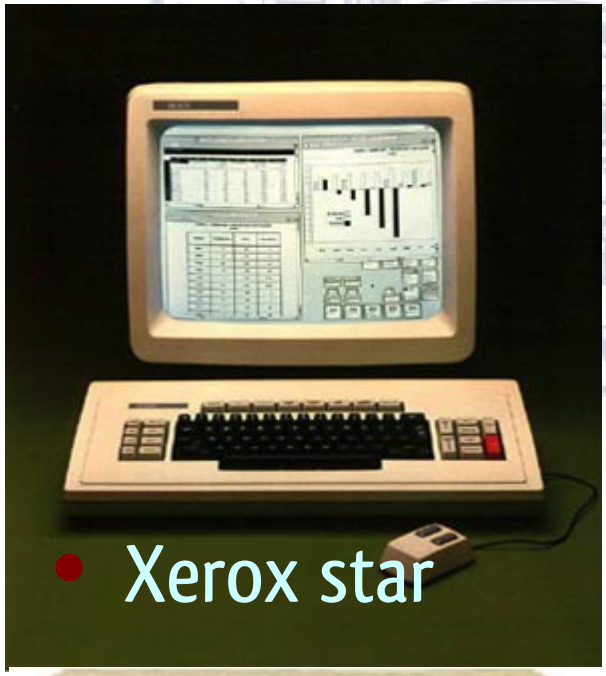
Storia

- 1961
  - Steve Russell at MIT crea il primo video game, Spacewar
- 1972
  - Nasce il videogioco Pong (Atari).
- Anche oggi una delle maggiori applicazioni della grafica interattiva è nel mondo dei videogiochi



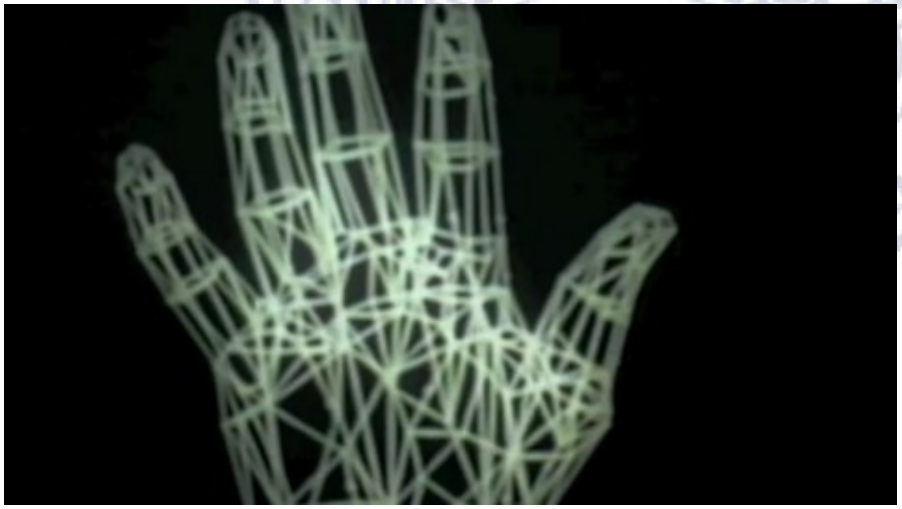
Storia

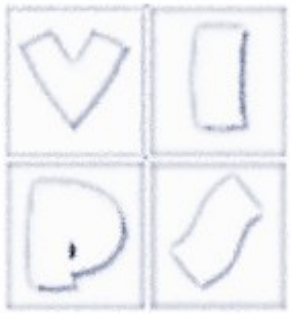
- Negli anni settanta nascono le moderne interfacce grafiche interattive dei computer (WIMP)
- La grafica interattiva, in questo caso 2D diventa parte integrante del sistema di interazione uomo-macchina



Storia

- Negli anni settanta nascono gli algoritmi per creare immagini da modelli 3D (rendering)
- 1972
  - Catmull (Univ. Utah) crea la prima animazione di grafica 3D
  - Modello della sua mano, 350 poligoni
  - Catmull diventerà un cofondatore della Pixar (oggi presidente)





Storia

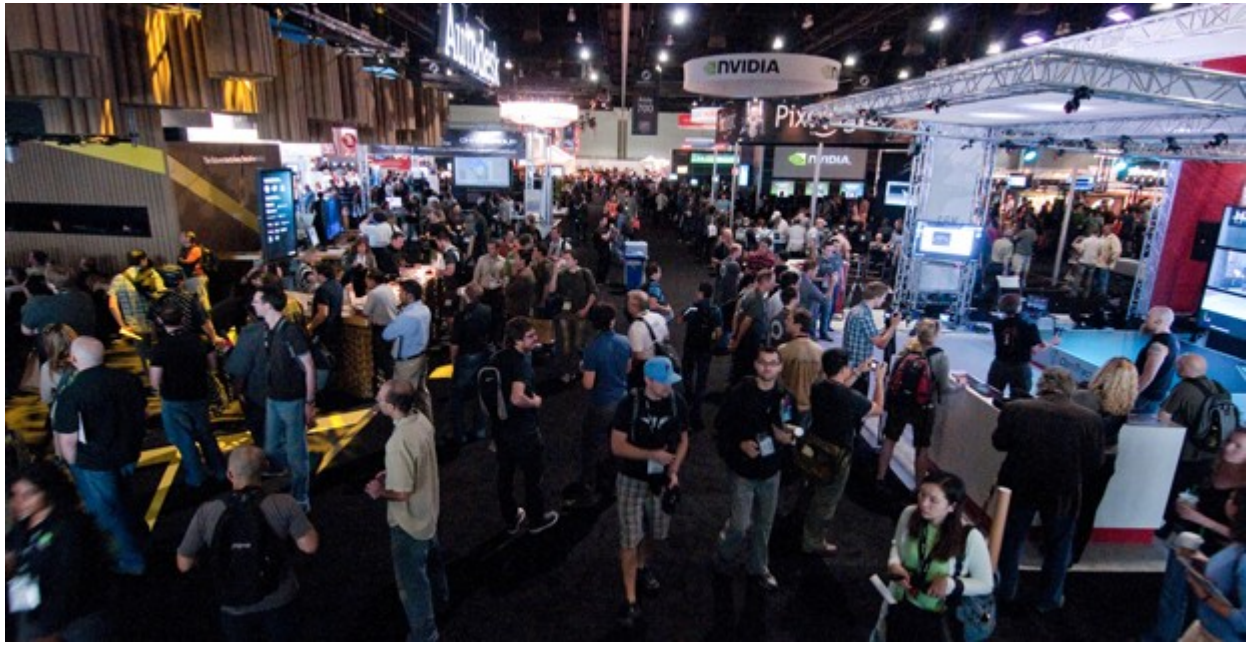
- Gli algoritmi per creare linee raster, riempire poligoni, proiettare oggetti 3D su telecamere virtuali vengono via via sviluppati negli anni '60-70-80
- Cuore della grafica 3D e di questo corso
- Si creano le pipeline di **rendering** per creare velocemente immagini sullo schermo a frame rate interattivi
- Si creano standard e implementazioni di sistemi grafici e si arriva alla situazione attuale
  - 1992 Silicon Graphics crea lo standard OpenGL
  - 1995 Microsoft rilascia Direct 3D
- La grafica ha pesantemente condizionato lo sviluppo dell'hardware e l'architettura dei moderni calcolatori (e tablet, smartphone, ...)

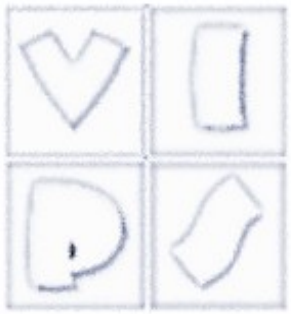
Storia

- Le operazioni grafiche vengono implementate su hardware specifico
- Inizialmente grafica raster calcolata su CPU, poi (doppio) buffer per mantenere le immagini (doppio perché il calcolo può essere lento rispetto al refresh dello schermo)
- 1985 Commodore Amiga, uno dei primi home computer con una GPU (Graphical Processing Unit)
- 1987 primo PC Ibm con operazioni 2D hardware
- 1995: prime schede video per PC con pipeline grafica 3D (S3 Virge)
- 1999 Nvidia GeForce 256 prima scheda con transform & lightning engine
- Vedremo meglio più avanti

Storia

- 1969, the ACM initiated A Special Interest Group in Graphics (SIGGRAPH)
- 1973 SIGGRAPH organizza la prima conferenza internazionale



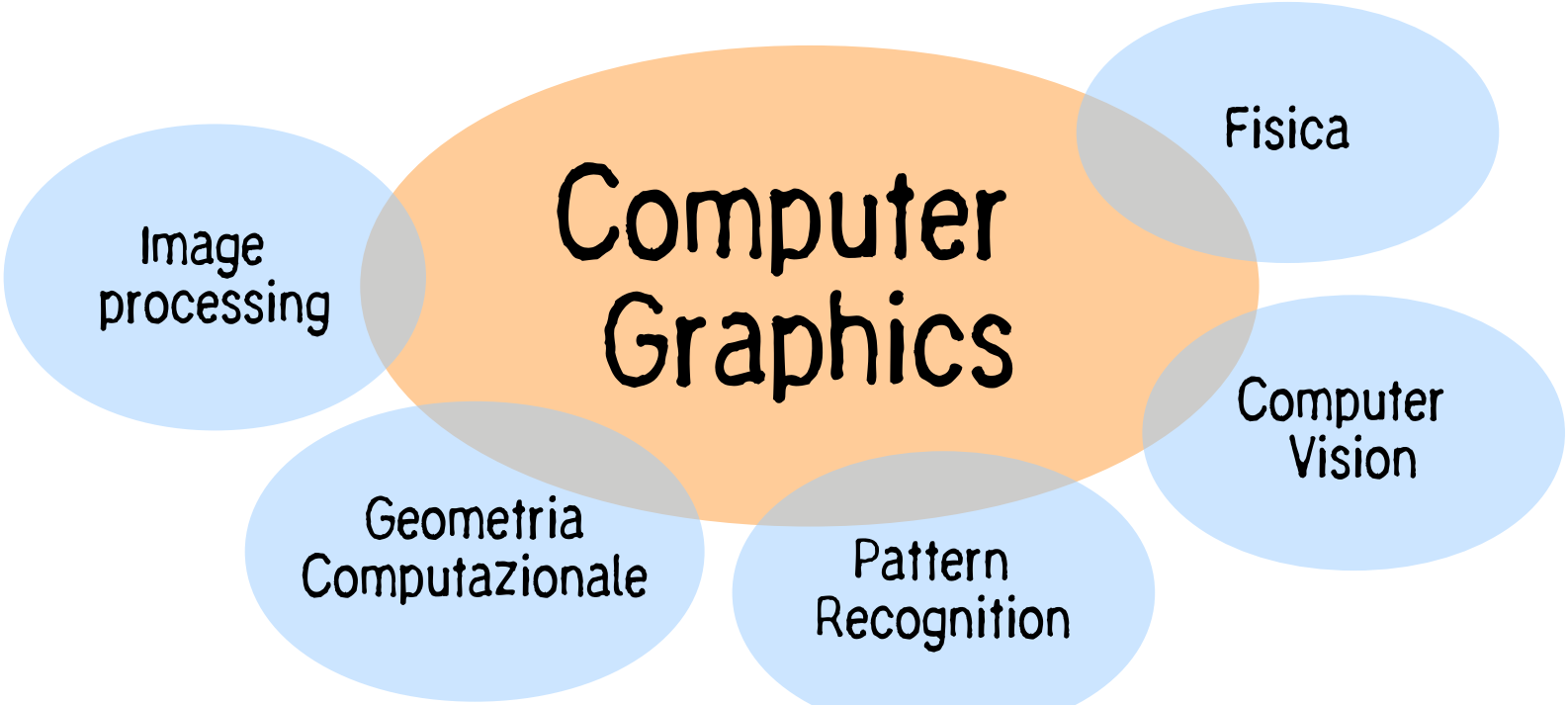


Quindi?

- Cos'è la “computer graphics”? Un po' tutto questo:
  - Creazione immagini 2d sintetiche e animazioni
  - Modellazione 2D, 3D, anche con comportamenti fisici
  - Computer Aided Design
  - Rendering delle scene, cioè creazione delle immagini simulando la proiezione ottica delle scene sulla camera
  - Animazione
  - Interfacce grafiche dei computer
  - Realtà virtuale
  - Enhancement video televisivo
  - Visualizzazione scientifica e dell'informazione

Una definizione semplice

- La disciplina che studia le tecniche e gli algoritmi per la rappresentazione visuale di informazioni numeriche prodotte o semplicemente elaborate dai computer (da Scateni e al.)
- E' quindi legata a molte altre discipline



Computer Graphics vs Computer Vision

- In senso generale la grafica è l'opposto dell' "image understanding" o della "computer vision"
  - Nel primo caso si passa da immagini a parametri, a interpretazione
  - Nel secondo si crea un'immagine da un input parametrico
  - Quindi sono grafica tutti i sistemi informatici che creano e usano immagini sintetiche

Visual Computing

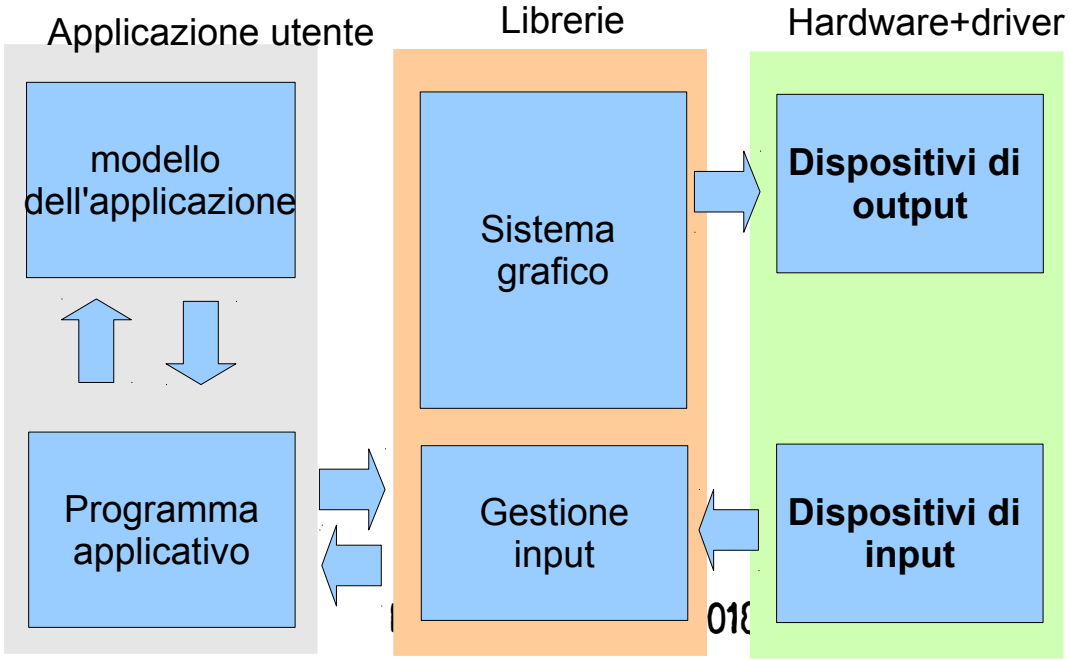
- Oggi vista la convergenza dei due domini si parla spesso in generale di “visual computing”
- **Visual computing** is a generic term for all computer science disciplines handling with images and 3D models, i.e. computer graphics, image processing, visualization, computer vision, virtual and augmented reality, video processing, but also includes aspects of pattern recognition, human computer interaction, machine learning and digital libraries. The core challenges are the acquisition, processing, analysis and rendering of visual information (mainly images and video). Application areas include industrial quality control, medical image processing and visualization, surveying, robotics, multimedia systems, virtual heritage, special effects in movies and television, and computer games.

Applicazioni della grafica

- Se lo scopo della grafica al calcolatore è quello di generare una rappresentazione visuale, quel che dovrà fare il nostro software è preparare i valori di output da passare al display in grado di generare lo stimolo per gli occhi umani
- Il display (output) può essere differente a seconda dell'applicazione.
  - In generale sarà un display raster come un monitor, che riproduce una matrice di punti su cui è codificata una terna di valori di colore RGB
- Ma sono dominio della grafica applicazioni che
  - Preparano file per la stampa 2D (grafica vettoriale)
  - Stampa 3D
  - Display innovativi (ad esempio stereo, volumetrici)

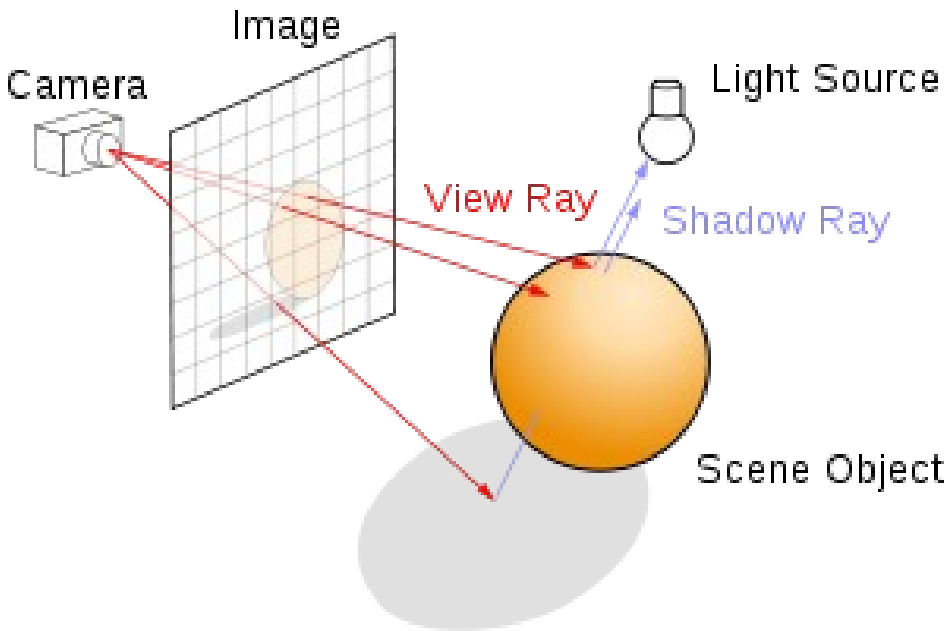
Grafica e videogiochi

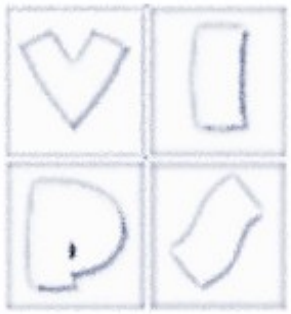
- La grafica dei videogiochi è di tipo interattivo, cioè deve generare una rappresentazione modificabile in tempo reale dall'utente
- Di solito 3D, ma non necessariamente
- Un'applicazione grafica interattiva ha generalmente una struttura come quella dello schema seguente e genera "immagini" a frame rate > 10 fps in funzione dell'input



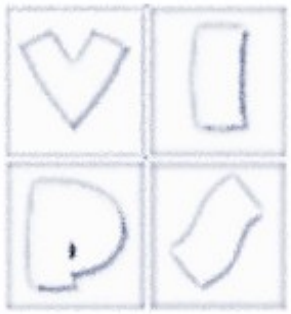
Ok cosa ci interessa capire

- Generare **immagini** per l'utente
- Da visualizzare **interattivamente** su **display**
- A partire da simulazioni di **ambienti fisici** (**geometria+materiali+luce**)
- Simulando la **formazione delle immagini reali**



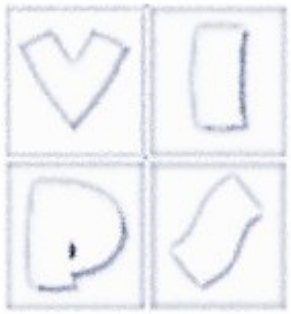


Cosa vogliamo generare e vedere: le immagini digitali



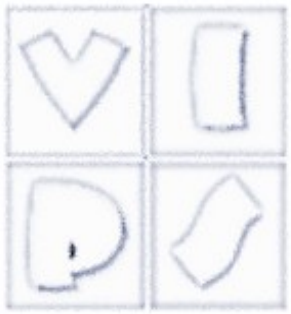
Grafica raster e vettoriale

- Quali “immagini” deve generare?
- In teoria possiamo rappresentare un disegno in un dato digitale in due modi
- Vettoriale
  - Primitive di disegno
- Raster
  - Griglia di valori da riprodurre sul monitor



Grafica vettoriale

- La rappresentazione grafica vettoriale compone le immagini come insieme di primitive di disegno
  - Linee
  - Curve
  - Aree
- Queste possono essere descritte con funzioni parametriche e coordinate di punti
- Applicazioni
  - Disegno su display vettoriali
  - Plotter
  - Rappresentazione per la stampa, necessita di conversione a raster di solito effettuata dalla stampante
  - Rappresentazione interna nei calcolatori di forme grafiche che devono essere rappresentate a differente livello di precisione
    - Es. caratteri di stampa (che facciamo grandi o piccoli, ecc.)



Grafica vettoriale

- Vantaggi
  - I dati sono espressi in una forma direttamente comprensibile ad un essere umano (es. lo standard SVG);
  - Compattezza di codifica rispetto all'equivalente raster;
  - Possibilità di ingrandire l'immagine arbitrariamente, senza che si verifichi una perdita di risoluzione dell'immagine stessa.
- Limiti:
  - Per la rappresentazione sulla maggior parte dei display occorre poi convertire a raster

Grafica vettoriale

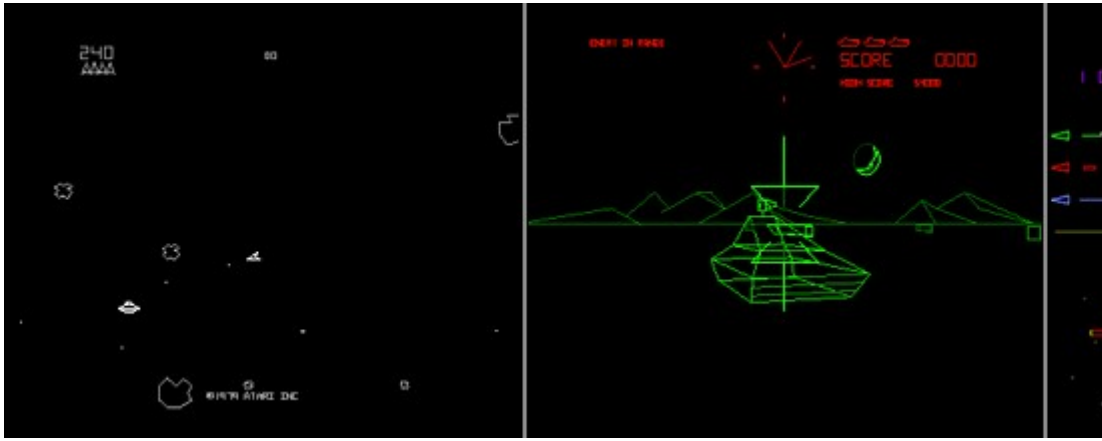
- Es Formato SVG


```
…
<rect id=”Rectangular_shape” fill=”#FF9900”
width=”85.302” height=”44.092”/>
<ellipse id=”Elliptical_shape” fill=”#FFFF3E”
cx=”42.651” cy=”22.046” rx=”35.447” ry=”16.426”/>
<text transform=”matrix(1 0 0 1 16.2104 32.2134)”
font-family=”ÅfMyriad-RomanÅf” font-
size=”31.2342”>SVG</text>
…
```

- Es Scalatura
  - Raster vs vector

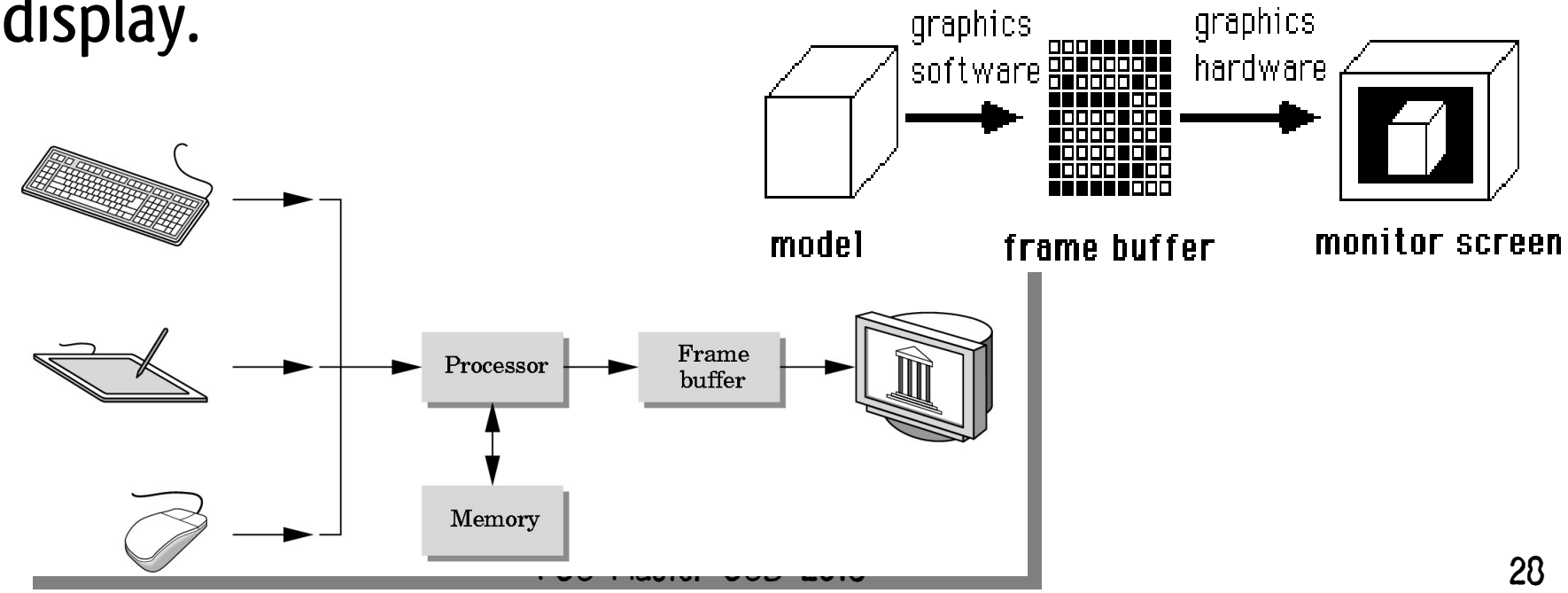
Display vettoriali

- Inizialmente (primi anni '60) molti dispositivi grafici di tipo vettoriale, in grado di tracciare direttamente linee e punti (stesso concetto dei plotter a penna)
- La grafica di quegli anni usava quindi primitive di disegno di tipo vettoriale e modalità di visualizzazione wire frame
- Anche famosi videogiochi (e.g. asteroids)



Immagini raster

- Le immagini digitali cui siamo abituati con i monitor attuali sono immagini **raster** e consistono di una matrice di elementi denominati **pixel**
- Il nostro software scriverà quindi un "frame buffer": memoria contenente l'immagine, array di valori per i pixel, che viene modificato direttamente dal programma di grafica video controller il quale legge il frame buffer e costruisce l'immagine sul display.



Immagini raster

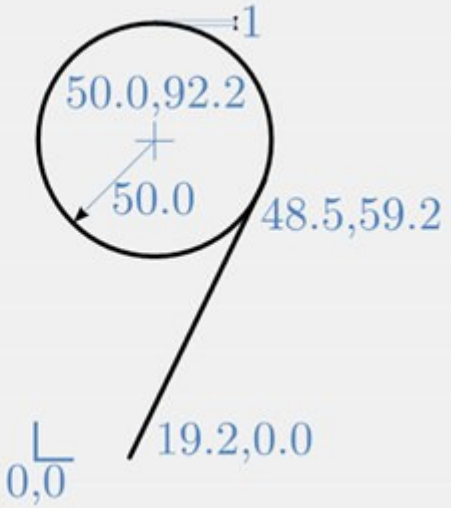
- Sono matrici che contengono valori che rappresentano il colore nella casella corrispondente agli indici con valori discretizzati
- Di solito componenti di colore: i monitor generano il colore con sovrapposizione di luce rossa, verde e blu (Perché?)
- Caratteristiche principali (non le uniche): risoluzione, (dimensioni della matrice di pixel), profondità di colore, (bit di memoria per pixel)
  - 8-bit significano 256 colori, mentre 24-bit (o truecolor) rappresentano all'incirca 32 milioni di colori
- Nota: è la rappresentazione di uscita tipica di quasi tutti i display odierni, ma non è ovviamente l'unica possibile
  - Es. display vettoriali: riproducono disegni
  - Il formato digitale creato internamente deve ovviamente corrispondere alla capacità del display scelto di riprodurlo

Immagini vettoriali e rasterizzazione

- La qualità della conversione dipende dalla risoluzione (numero di punti o punti per pollice)
- Scalettatura: effetto dell'aliasing delle alte frequenze
- Si può ridurre l'effetto sfumando la luminosità (antialiasing)
- La conversione è stata una delle prime cose ad essere delegata all'hardware grafico.

```
NUMBER_OF_PRIMITIVES 2
CIRCLE
center 50.0,92.2
radius 50.0
fill_color white
line_color black
line_thickness 1pt

SEGMENT
FIRST_ENDPOINT 19.2,0.0
SECOND_ENDPOINT 48.5,59.2
line_color black
line_thickness 1pt
```

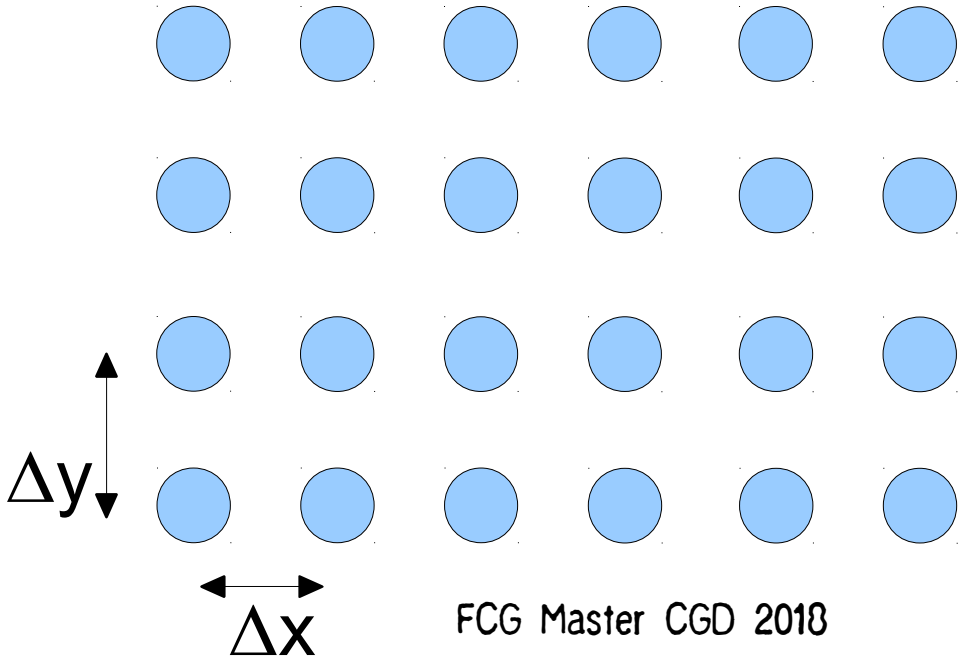


Immagini raster

- L'aliasing accade perché l'immagine è il campionamento di un segnale "continuo" che rappresenterebbe il dato misurabile

$$I(i,j) = I(i\Delta x, j\Delta y)$$

$$= \iint F(x,y)\,\delta(x - i\Delta x)\,\delta(y - i\Delta y)$$



Aliasing

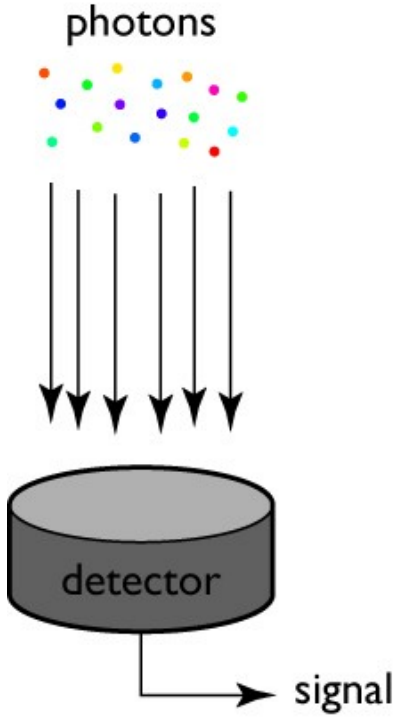
- Per il teorema di Shannon/Nyqvist del campionamento, non si può ricostruire esattamente il segnale originale se la frequenza del segnale è superiore alla metà della frequenza di campionamento

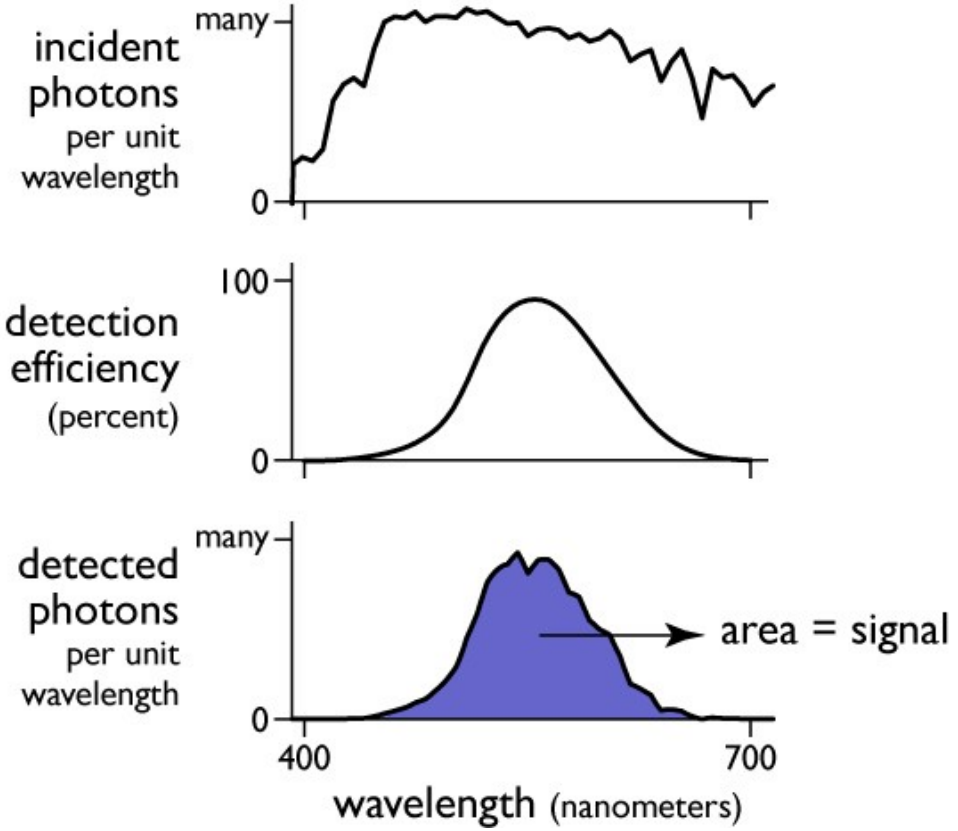
$$\nu_{cx} = \frac{1}{\delta x} \geq 2\nu_{x\max} \qquad \nu_{cy} = \frac{1}{\delta y} \geq 2\nu_{y\max}$$

- Dove ci sono discontinuità del colore, ci sono componenti a frequenza alta, si creano artefatti
- I filtri che fanno antialiasing attenuano le alte frequenze
- Le procedure di rasterizzazione possono usare metodi per ridurre l'aliasing

Sensori e luce

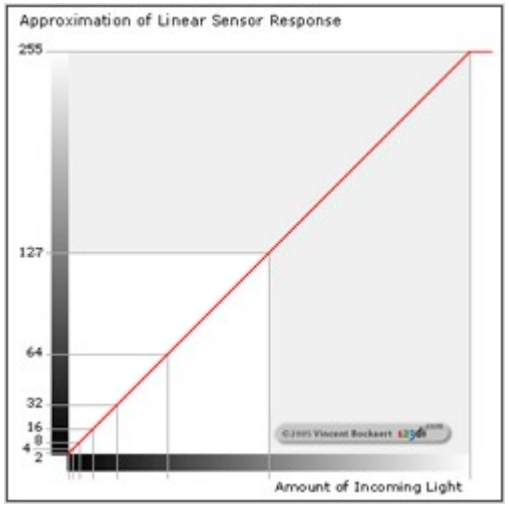
- Nella grafica 3D vogliamo simulare immagini di telecamere che misurano la luce che arriva su un piano
- Quello che misurano i sensori della retina e delle telecamere è l'energia che arriva al sensore dai fotoni. Ma i sensori sono sensibili solo a un intervallo di frequenze, in modo non uniforme

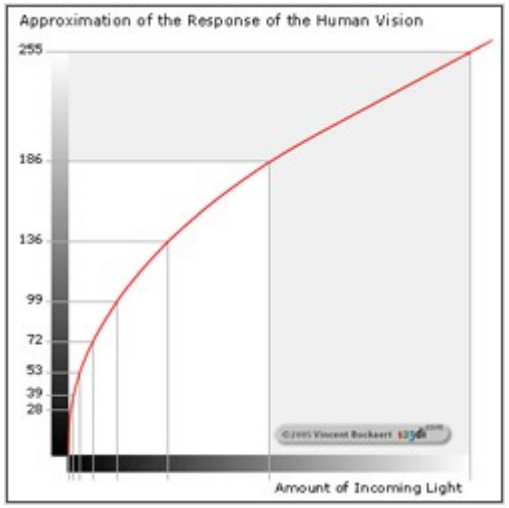


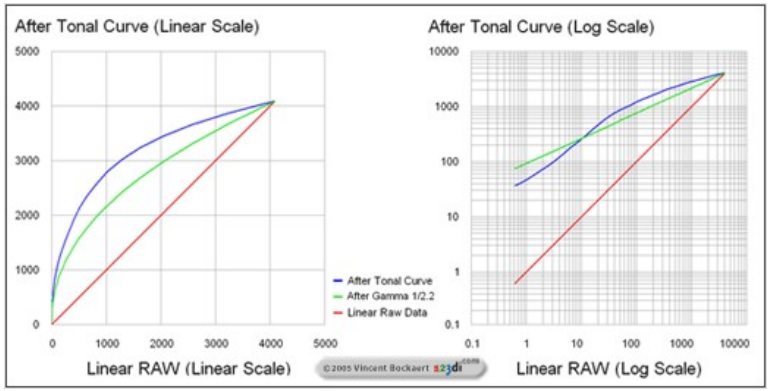


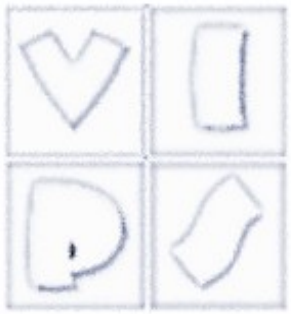
Percezione range dinamico

- La misura è tra l'altro in genere mappata in modo non lineare per rendere le differenze tra livelli discretizzati uniformememte percettibili
- La percezione umana amplifica le differenze ai bassi livelli



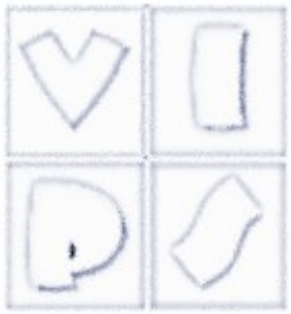






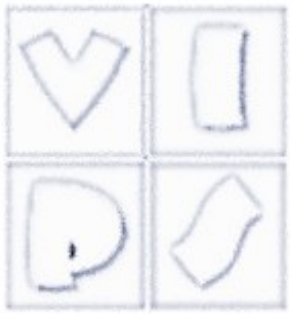
Colore

- Le immagini raster che genera la grafica al calcolatore sappiamo che sono, in genere, a colori, per simulare il modo in cui vediamo il mondo (a colori)
- La rappresentazione del colore è generalmente una terna di valori RGB
- Per la riproduzione corrispondono alle intensità emesse da tre emettitori di luce a tre frequenze determinate (rosso, verde, blu) che danno origine in corrispondenza a un certo colore percepito dall'utente
- Ma cosa significa questo? Che cos'è il colore?



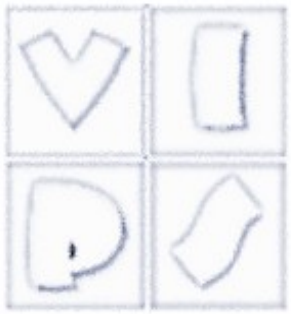
Immagini e realtà

- Con la grafica fotorealistica al calcolatore vorremmo in teoria riprodurre la formazione delle immagini sul sensore della telecamera o sull'occhio
- Punto fondamentale: le immagini digitali NON riproducono e non possono riprodurre la realtà
  - Discretizzazione spaziale e temporale
  - Uso di frequenze discrete nello spettro elettromagnetico
- Quello che conta è riprodurre la sensazione al cervello umano
- Quindi è fondamentale conoscere come funziona la percezione umana



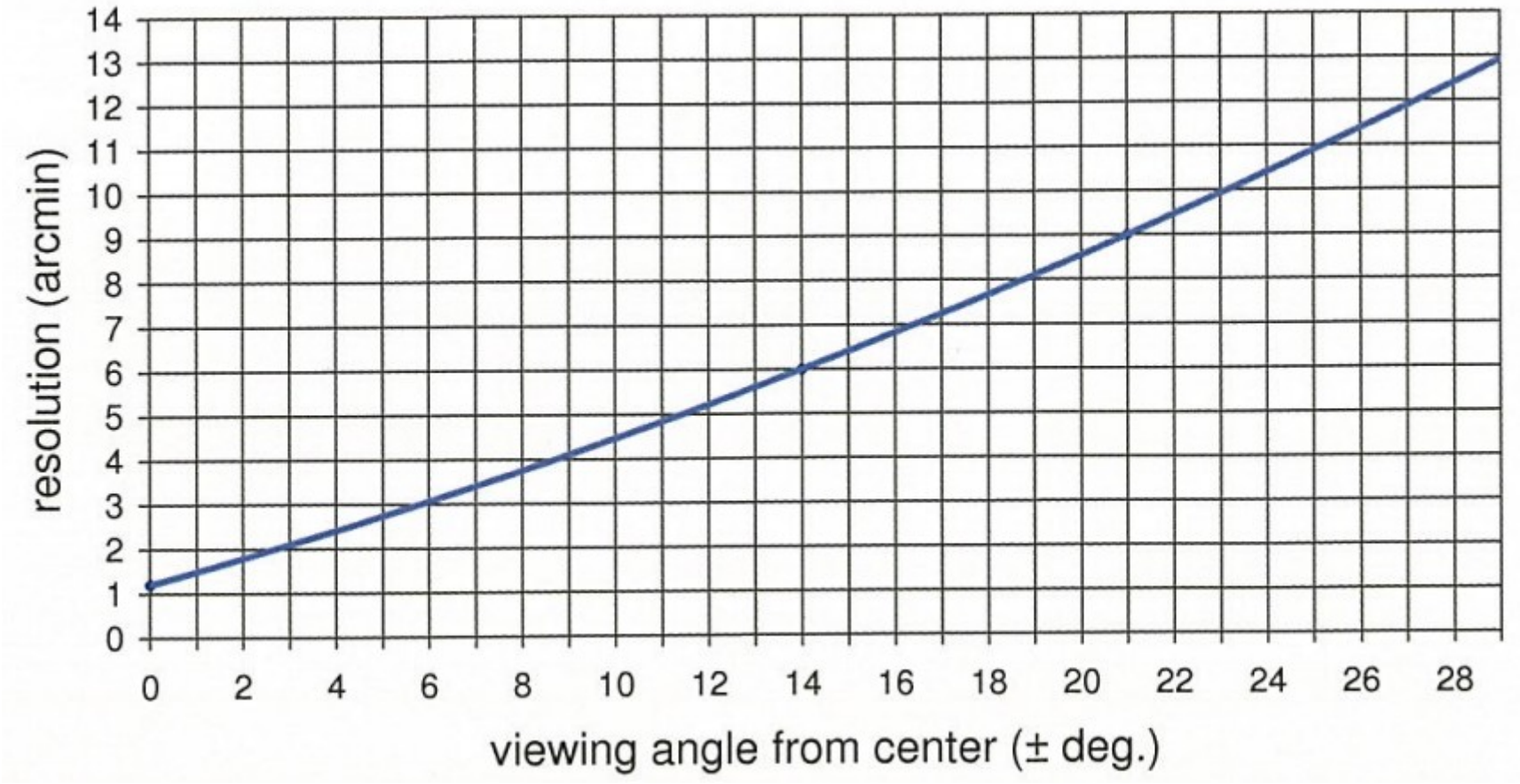
Human Vision System

- visual acuity: 20/20 is ~1 arc min
- field of view: ~200˚ monocular, ~120˚ binocular, ~135˚ vertical
- resolution of eye: ~576 megapixels
- temporal resolution: ~60 Hz (depends on contrast, luminance)
- dynamic range: instantaneous 6.5 f-stops, adapt to 46.5 f-stops
- colour: everything in CIE xy diagram
- depth cues in 3D displays: vergence, focus, (dis)comfort
- accommodation range: ~8cm to ∞, degrades with age



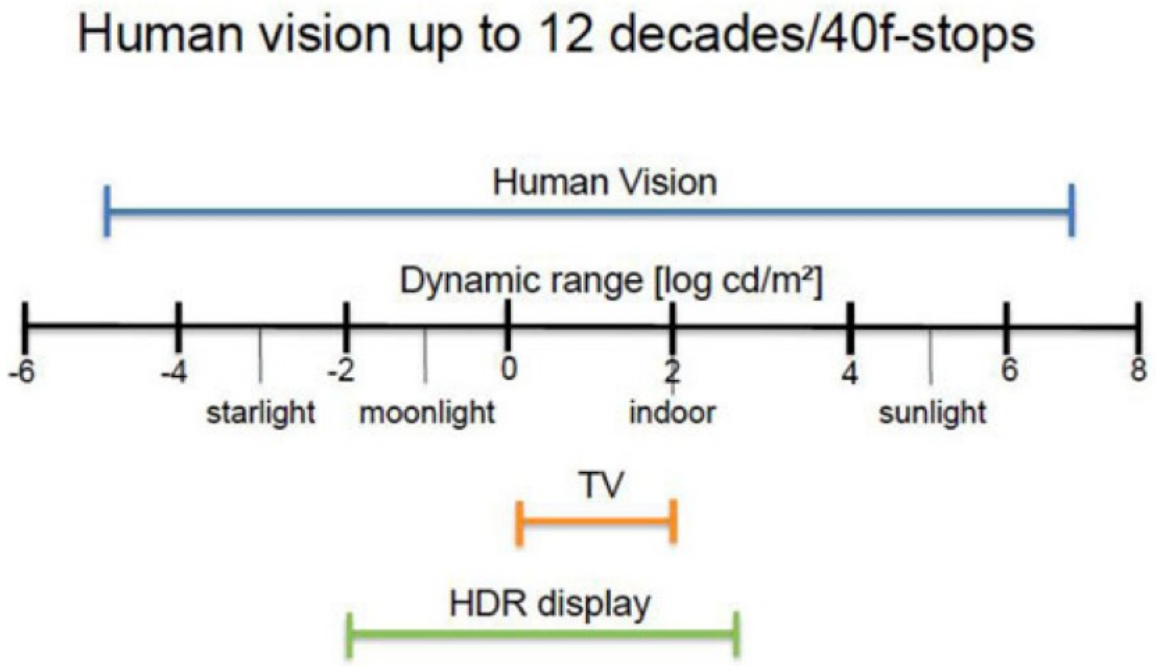
Resolution

- Decrease far from fovea



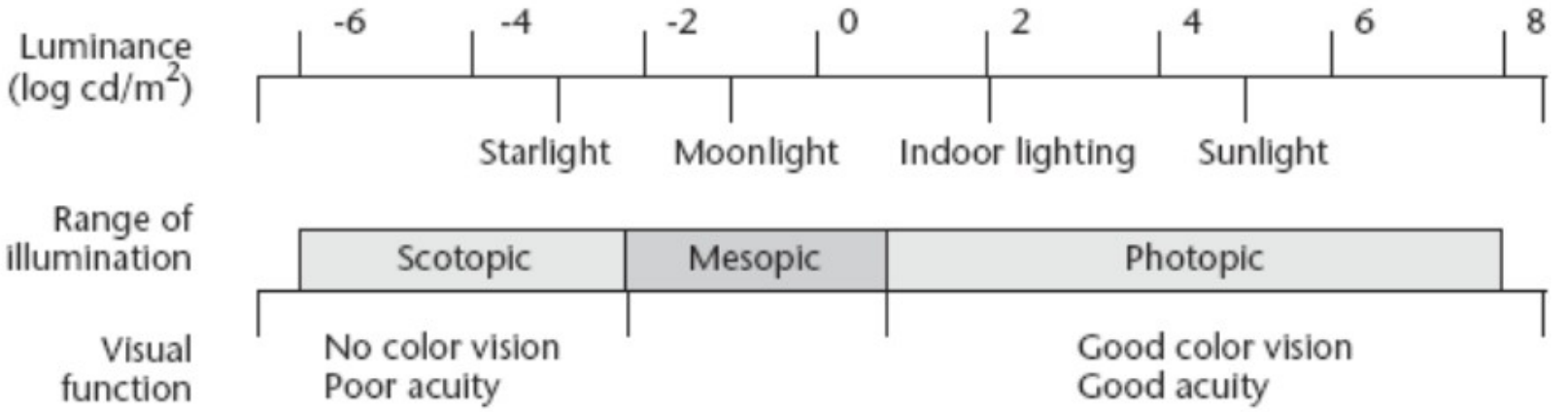
Dynamic range

- Human vision has far higher dynamic range than any available display technology
  - 40 f-stops, cf 17 f-stops for HDR display



Vision changes

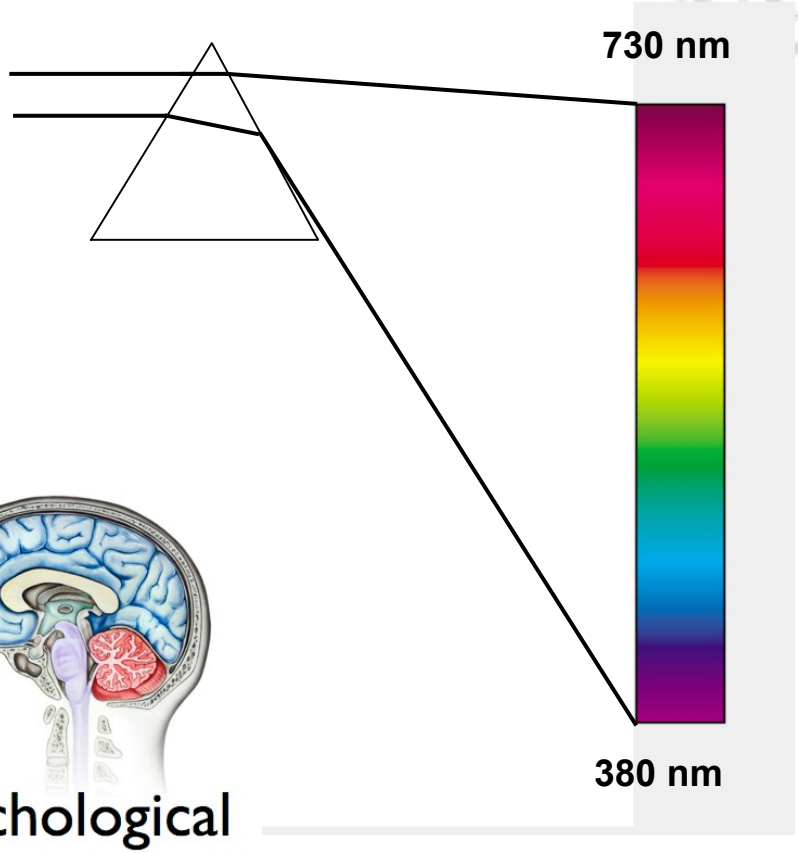
- With color and luminance
- Colors are percepted only with high luminance

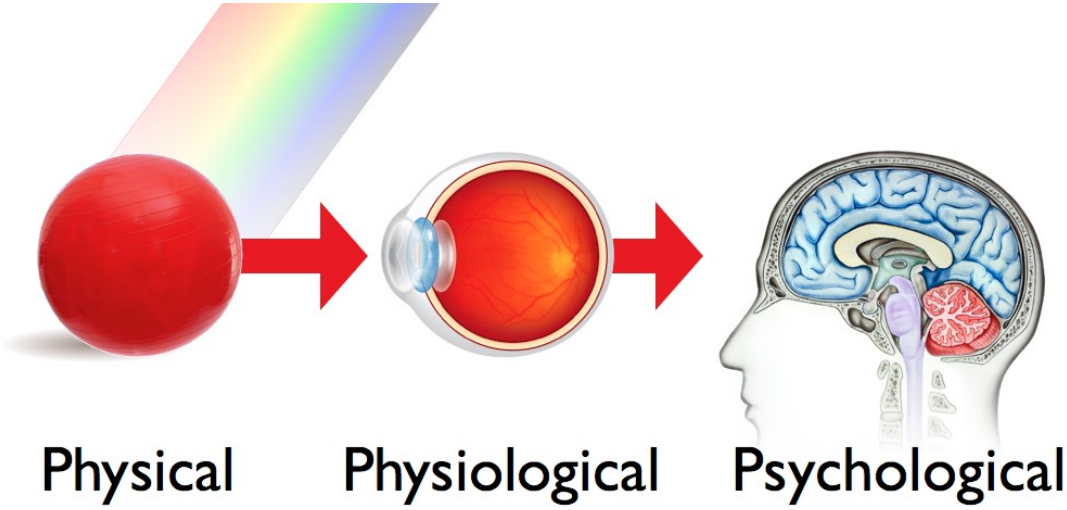


[Ferwerda 2003]

Colore

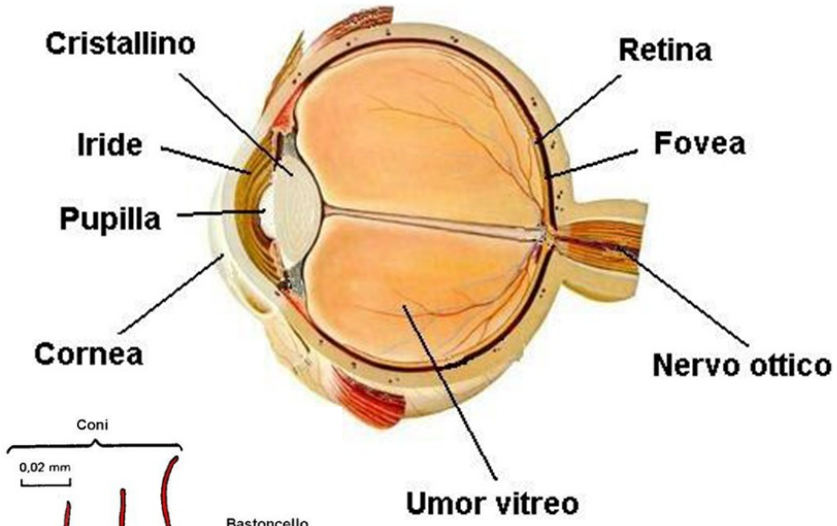
- Non esiste nel mondo
- La luce è un continuo di frequenze
- E' una caratteristica percettiva dell'uomo che dipende
  - Dal funzionamento dei recettori
  - Dall'elaborazione del cervello

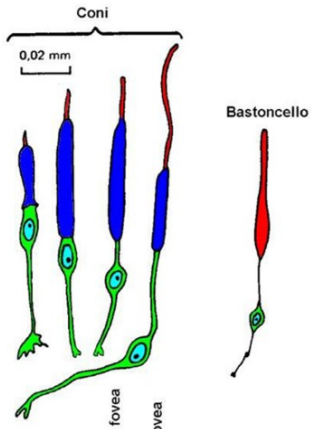


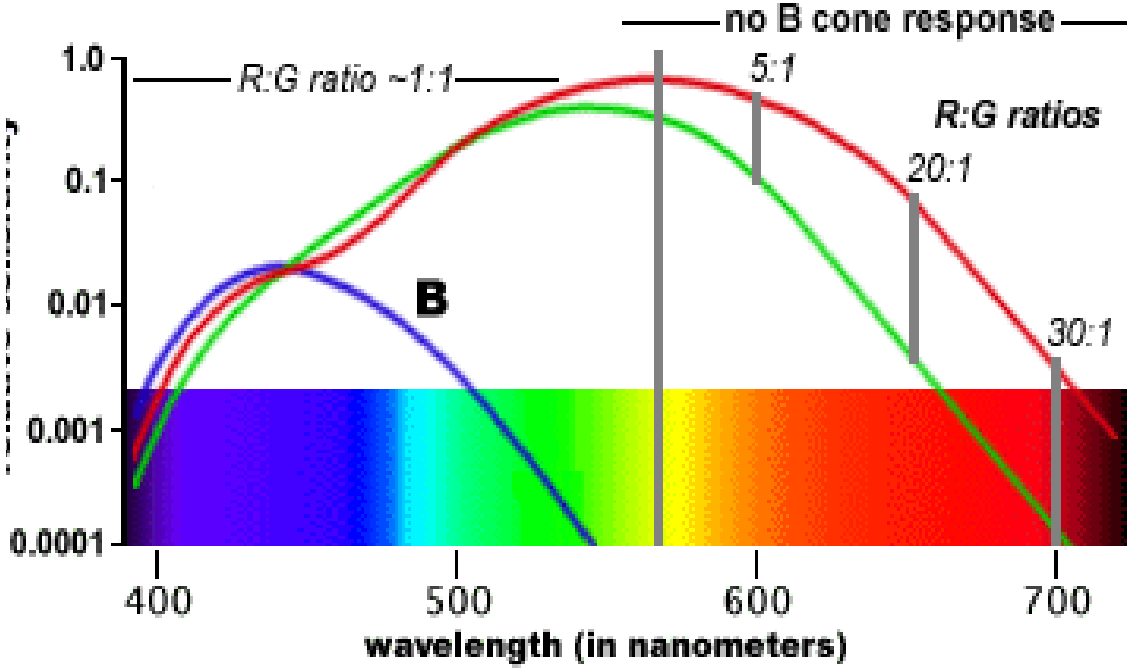


Percezione del colore

- Percezione del colore: nella retina ci sono 3 tipi di coni, che hanno differenti sensitività a diverse frequenze S,M,L
- Possiamo fare il matching delle frequenze con la risposta dei recettori

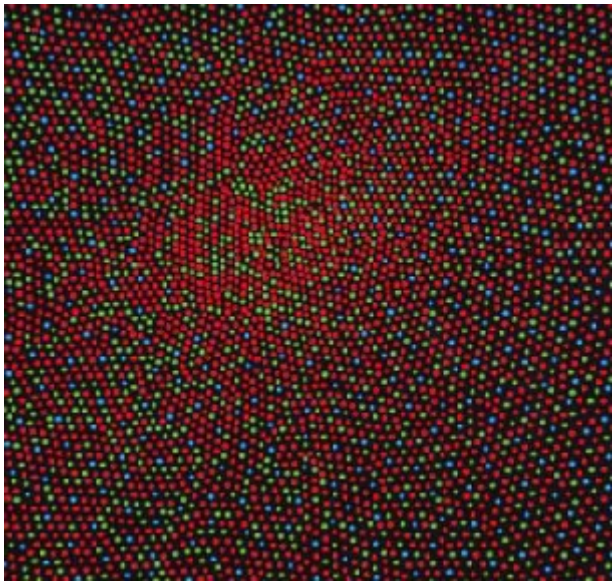


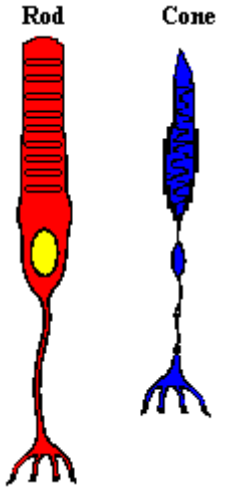




Recettori e percezione

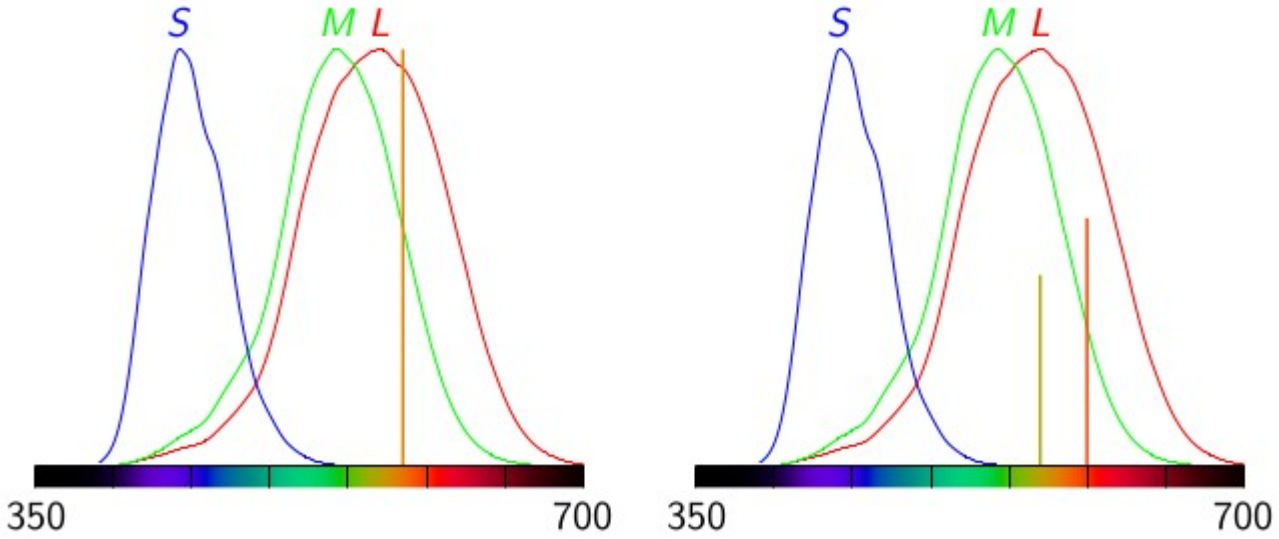
- I recettori sono di due tipi
- Bastoncelli, situati maggiormente nelle zone periferiche del campo visivo
  - Rispondono a basse luminosità (1000 volte più dei bastoncelli)
  - Detezione contrasto e movimento. Singolo range di frequenze (no visione a colori)
- Coni addensati intorno alla fovea
  - Tre tipi di risposta in frequenze (S,M,L)
  - Concentrati al centro
  - Meno sensori S



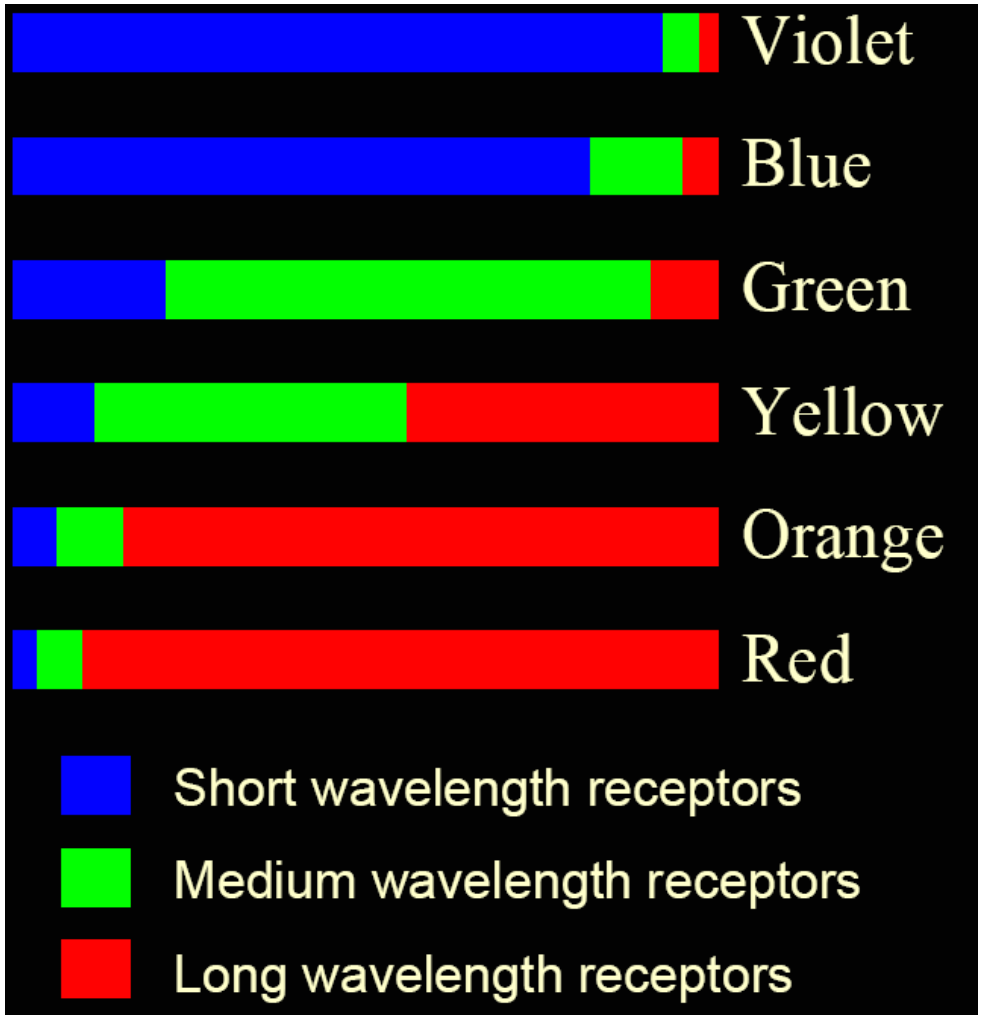


Visione e immagini a colori

- Il "colore" percepito a livello fisiologico è dato da 3 grandezze scalari, funzione dello spettro della luce incidente.
- La corrispondenza non è iniettiva. Spettri diversi possono corrispondere allo stesso colore percepito: metamerismo
- Conseguenza: Per riprodurre un colore, non è necessario riprodurre lo spettro. È sufficiente che le risposte L, M, S dei coni siano uguali
- Può cambiare in funzione dell'illuminazione



Risposte dei coni



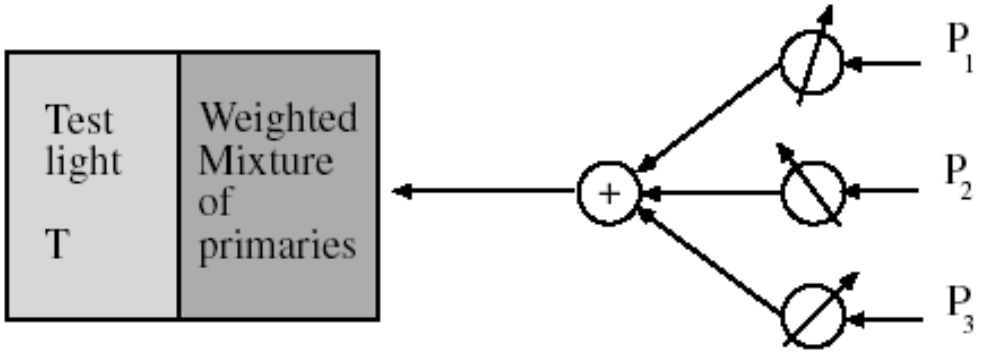
- Ogni colore è caratterizzato dalle risposte dei diversi recettori (3 numeri)
- Da questi ricaviamo le sensazioni di "colore"
- Poi modificate dal processing cerebrale

Teoria tricromatica

- Legge di Grassman: l'uomo è in grado di fare match di colori mischiando 3 (o più) colori detti primari. Se la luce test T ha un certo colore,

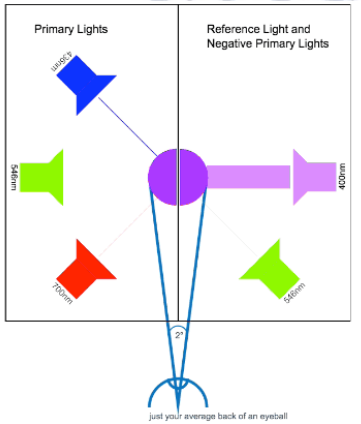
$$T=aP1+bP2+cP3$$

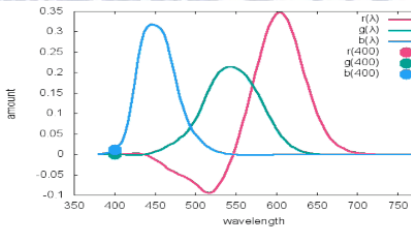
- il match si verifica lineare *(legge di Grassman)*



Tricromatismo

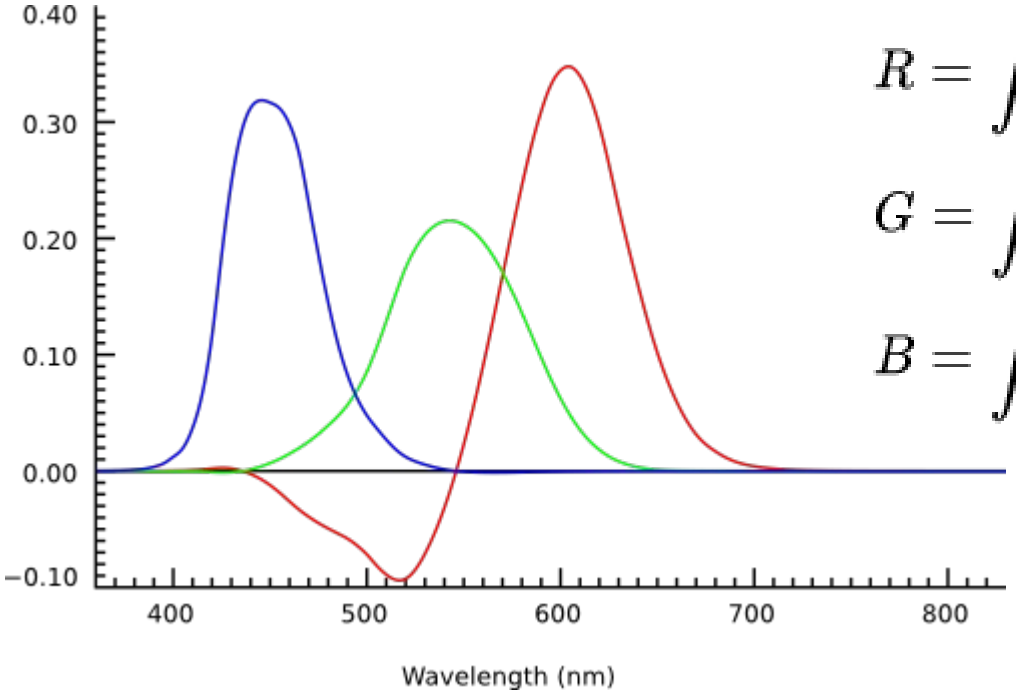
- Così come la percezione deriva dalle tre risposte dei coni, possiamo simulare la percezione del colore combinando luci di tre colori
- Colori primari, spazio di colore
- Quali scegliere? Esempio RGB, monitor
- Device dependent
- Problemi
  - Le risposte non sono ortogonali
- Per alcuni matching coefficienti negativi (colori non rappresentabili)
  - Negli esperimenti con utenti si può fare il matching mischiando il primario alla luce target





Funzioni di matching

- Data una terna di colori primari possiamo studiare il matching dei colori sugli osservatori in funzione della frequenza
  - Esperimenti che ricavarono “funzioni di matching”
- Componenti colore CIERGB ricavate dall'integrale delle frequenze dello stimolo su tutto il range



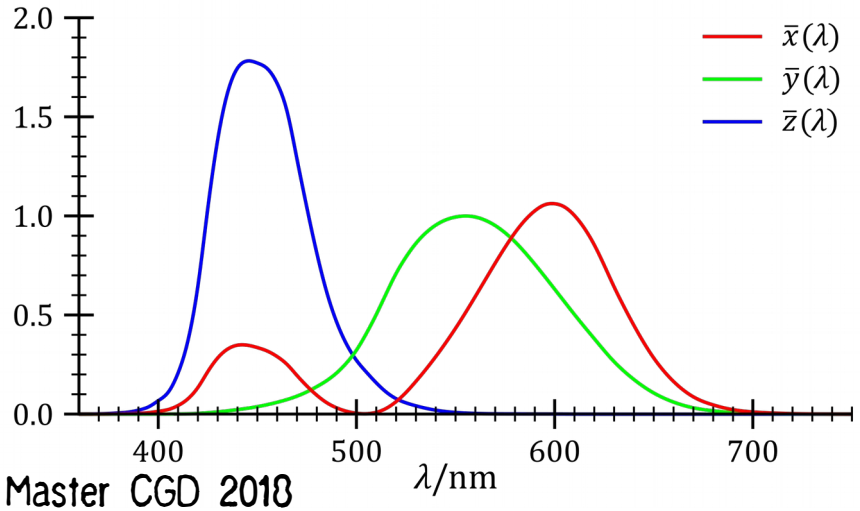
$$R = \int_0^\infty S(\lambda)\,\bar{r}(\lambda)\,d\lambda,$$

$$G = \int_0^\infty S(\lambda)\,\bar{g}(\lambda)\,d\lambda,$$

$$B = \int_0^\infty S(\lambda)\,\bar{b}(\lambda)\,d\lambda.$$

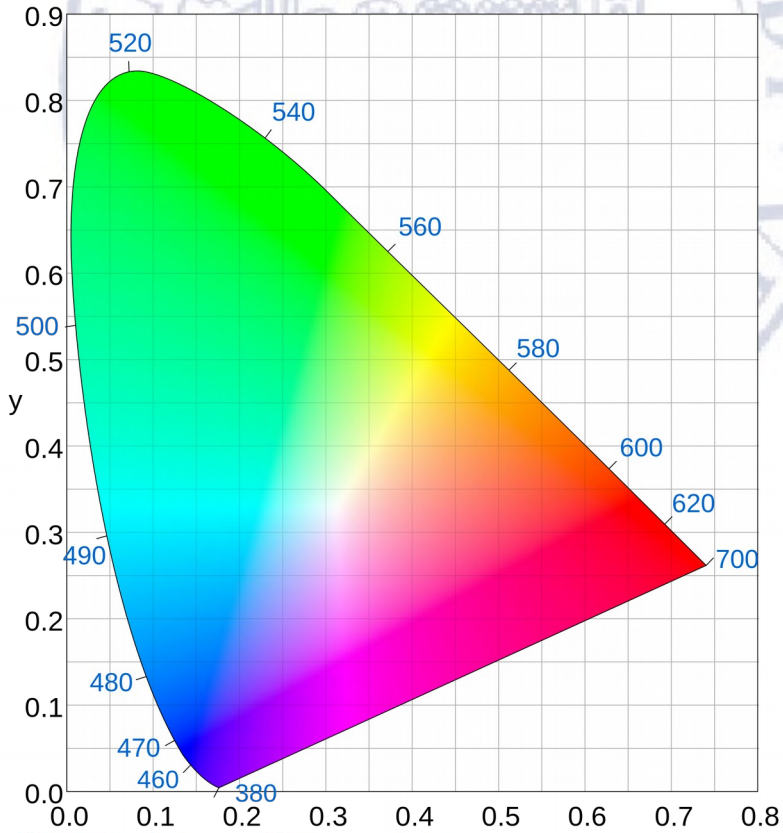
Spazio colore CIE XYZ

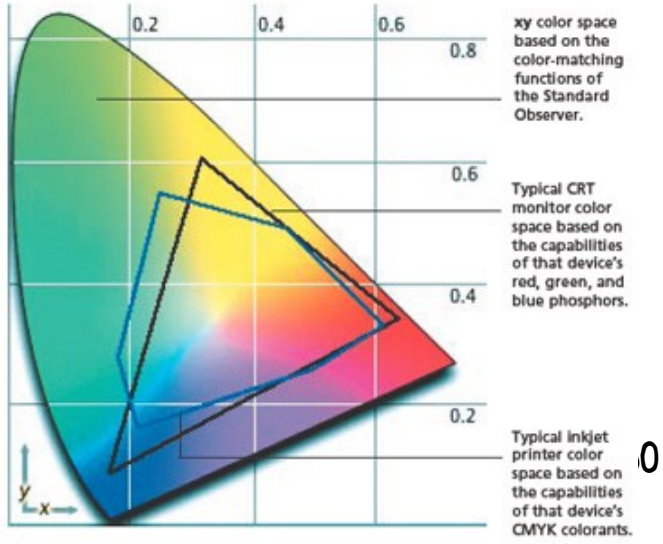
- Necessità di definire una misura del colore indipendente dal sistema di visualizzazione.
- Nel 1931 la Commission Internationale de l'Eclairage (CIE)
- propone lo standard XYZ.
- Idea: Rappresentare il colore mediante le risposte dei coni
  - Si stimano con esperimenti le risposte
  - Trasformazione spazio CIERGB tale che le componenti siano sempre positive

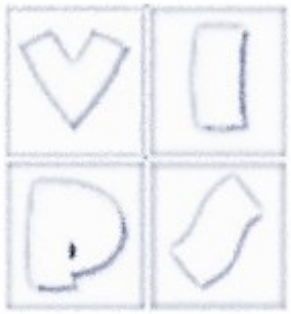


Colori visibili

- Diagramma cromaticità ottenuto dalle coordinate XYZ normalizzate (somma=1, prendo x,y)
- Rappresenta i colori visibili
- È uno spazio assoluto, non dipende dalla riproduzione
- RGB sì.
- Quali colori rappresenta un monitor RGB dipende dalla mappatura su XYZ





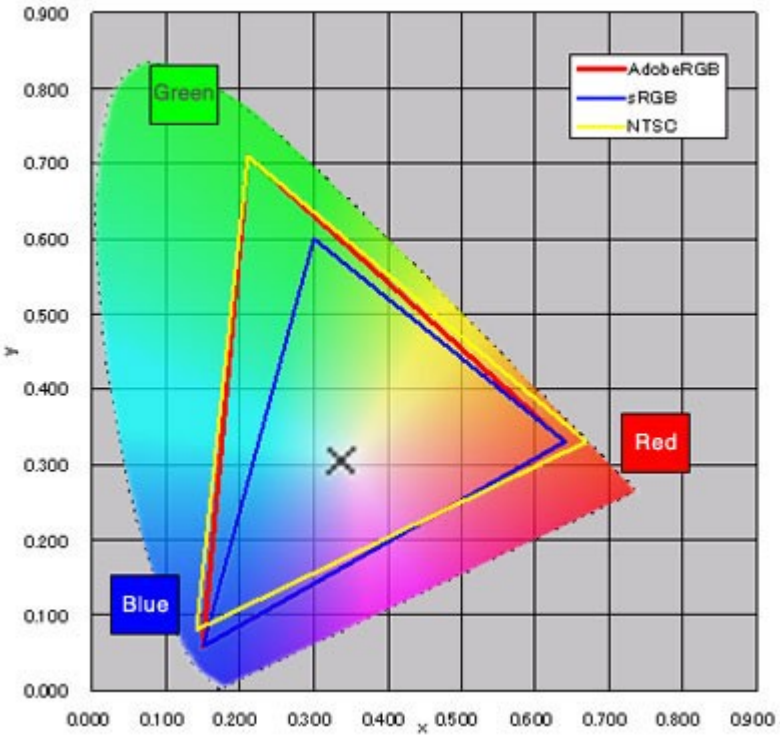


sRGB

- Una rappresentazione dello spazio colore indipendente da device
- Rappresentazione per device standard
- Simulazione del colore sui dispositivi ottenuta per calibrazione

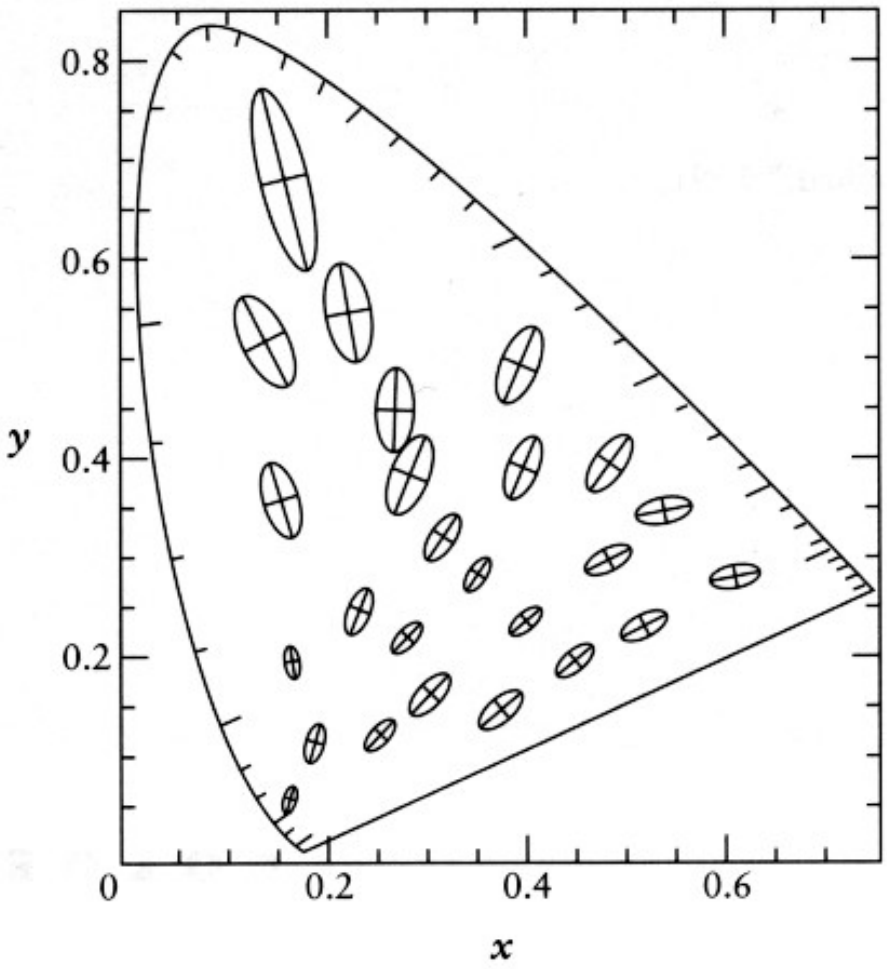
Gamut

- Per gamut o gamma si intende la gama di colori riproducibili da parte di un determinato dispositivo (o tecnica)
- Oppure dato un determinato modello di colore
- Per un display rgb è un triangolo coi vertici sui colori "primari" emessi dai pixel
- Il punto di bianco dice la cromaticità corrispondente a RGB massimi



Spazio CIELab

- Gli spazi visti non sono "percettivamente" uniformi
  - Ad uguali variazioni di colore, intese come uguali spostamenti all'interno dello spazio colore (anche per *X Y Z*), non corrispondono uguali differenze nella percezione degli stessi.
- Il nostro sistema visivo è sensibile in modo diverso alle varie lunghezze d'onda
- I recettori del blu sono molto meno sensibili
- Per avere uno spazio colore percettivamente uniforme, è stato definito dalla CIE nel 1976 lo spazio L*,a*,b*



Sorgenti luminose e materiali

- Notare che i colori "visti" non sono intrinseci degli oggetti e dipendono dalla combinazione di sorgenti luminose e riflettanze che variano da sorgente a sorgente e da materiale a materiale
- Al variare del tipo di sorgente luminosa gli spettri riflessi dagli oggetti cambiano completamente

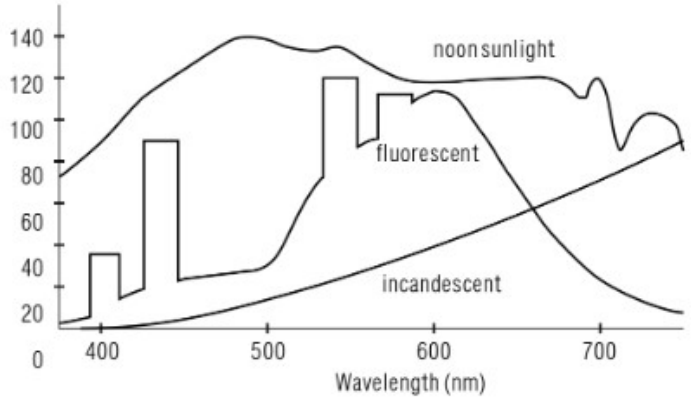


Figure 4.6: The *spectral power distribution* for some common light sources. (Figure from [292]).

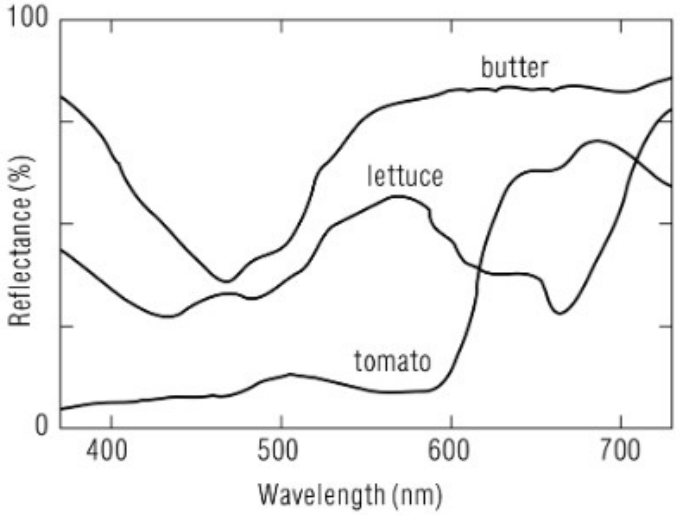
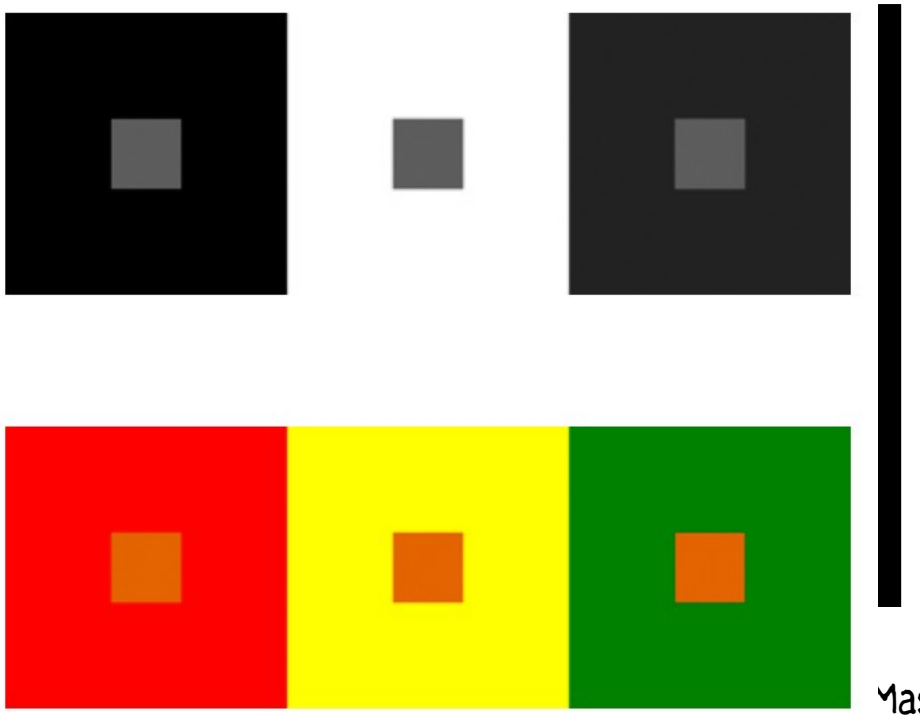
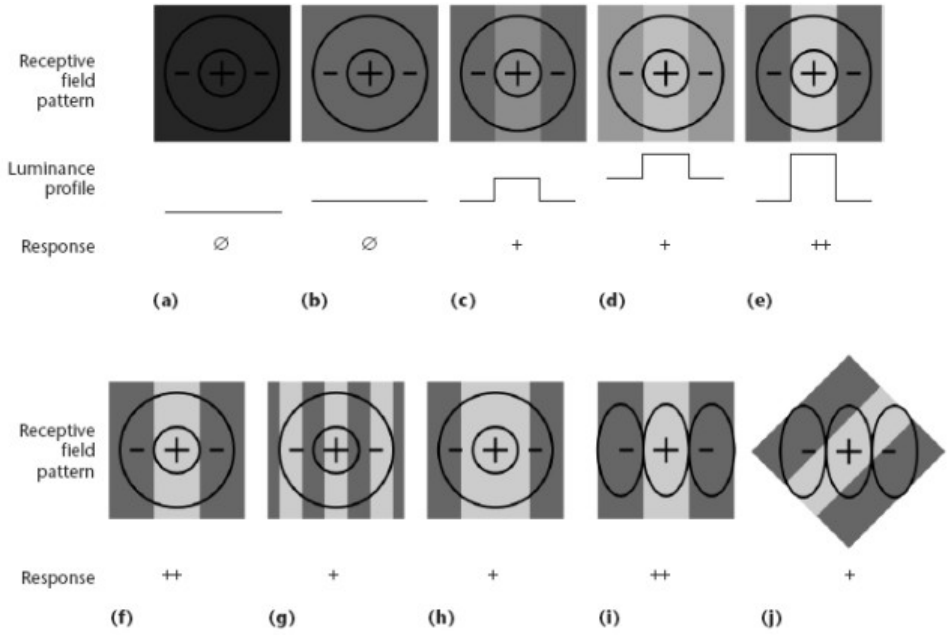


Figure 4.7: The *spectral reflection function* of some common familiar materials. (Figure from [292]).

Non basta il tricromatismo

- La percezione del colore è mediata dal cervello
  - Ci sono altri fenomeni che complicano la vita
- Non percepiamo il livello ma il contrasto
- Diversa impressione a seconda dell'intorno

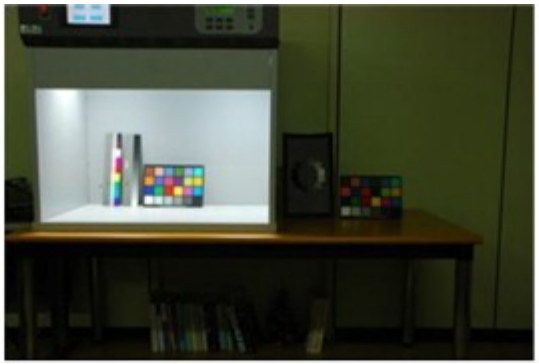




Master C

Adattamento

- luminance adaptation
  - Riscaliamo mentalmente la luminosità adattandoci a livelli variabili (es al chuso, a teatro)
- chromatic adaptation
  - Adattiamo la percezione del colore al colore della sorgente luminosa
  - Come il bilanciamento del bianco nelle fotocamere
  - In ambienti con lucu colorate (es. discoteca) dovremmo vedere colori completamente diversi, ma non ce ne rendiamo conto



Kwon et al.

HDR image rendering

- Si fa qualcosa di simile nelle tecniche di tone mapping per le immagini HDR, dove si acquisisce il segnale con un alto range dinamico poi si rimappa adattativamente spesso simulando il processing dell'uomo

Daltonismo

- I recettori possono funzionare male...

Normal

Protanopia

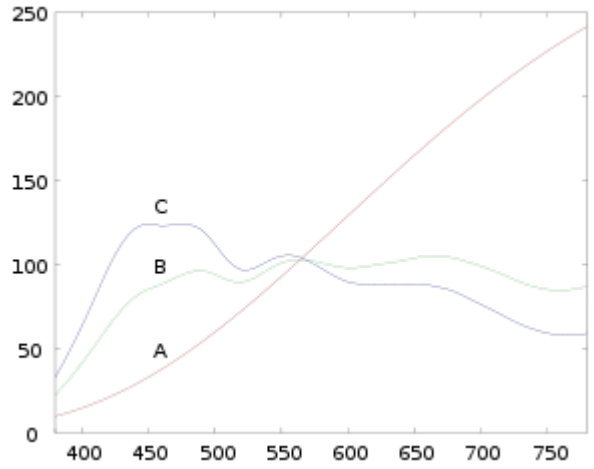
Deuteranopia

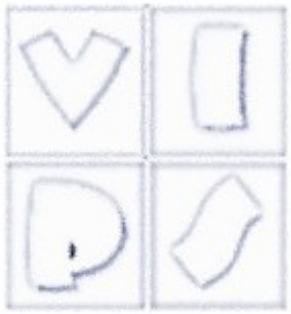
Tritanopia

RGB e sensori

Nelle immagini RGB acquisite da sensori, vengono registrate nelle componenti le energie acquisite nello spettro di acquisizione dei sensori

- I valori dipendono dal sensore e dalla distribuzione spettrale del bianco di riferimento
- Inoltre può essere applicata anche una correzione della non linearità



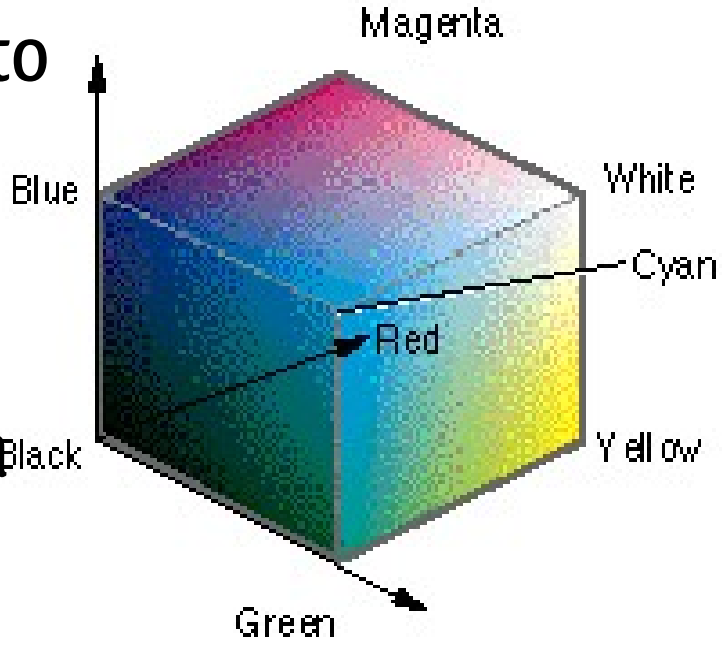


RGB e display

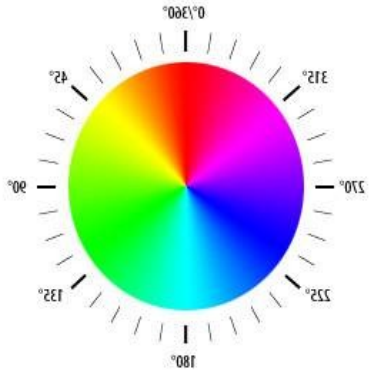
- Il colore realmente generato da un display data la terna RGB dipende da
  - La risposta spettrale degli emettitori
  - La calibrazione del monitor (punto di bianco)
  - La non linearità della risposta
- Per far corrispondere le coordinate RGB a un colore rappresentato dal monitor sul diagramma di cromaticità, dobbiamo mappare opportunamente

Spazi di colore

- Gli spazi che utilizziamo tipicamente in grafica sono quelli legati ai device di acquisizione e display
- "cubo" RGB dove rappresentiamo il colore acquisito dai sensori o generato additivamente dai monitor
- Per far corrispondere a un valore assoluto dovremmo poi avere la calibrazione di sensore o monitor
- Rappresenta e riproduciamo solo una parte dei colori visibili
- Poco intuitiva l'interpretazione delle caratteristiche del colore (tinta, chiaro, scuro) per questo si usano spazi alternativi



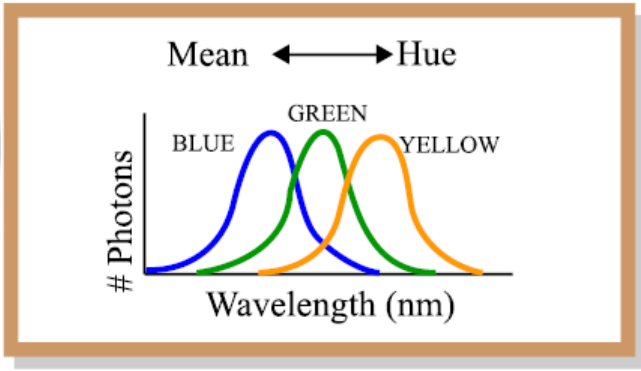
Spazio HSV

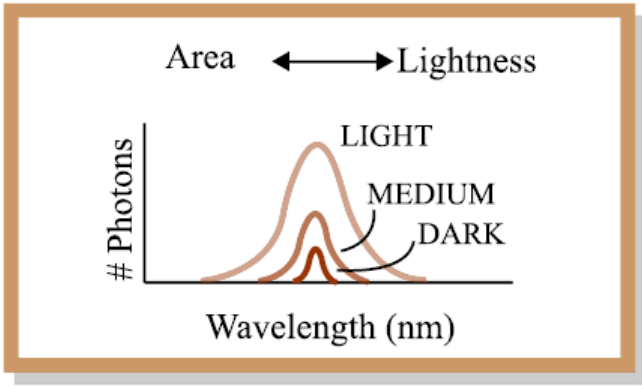


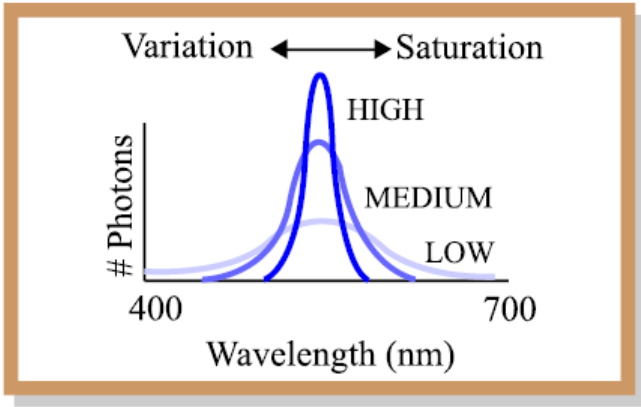
Hue: è la tinta vera e propria, il "colore" che percepiamo.

Saturazione: la distanza del colore dal grigio più vicino

Valore o illuminazione: si parla di quantità di luce o quantità di bianco di un colore

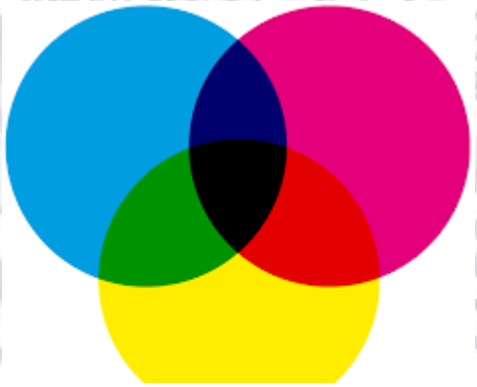


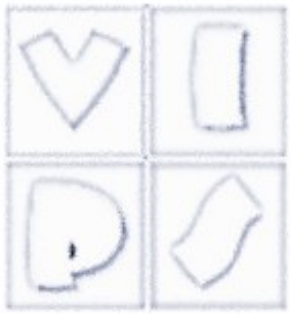




Spazi RGB e CMYK

- Allo stesso modo per rappresentare i colori “stampati” si usa lo spazio CMYK
- Cyan, Magenta, Yellow “complementari” di Red Green e Blue. Usando il nero per la parte uguale di CMY si ottiene miglior risultato
- Gli spazi RGB e CMYK sono stati concepi in funzione dei dispositivi di visualizzazione (monitor e stampanti), n non rispecchiano affatto la nostra percezione.
- Un modello alternativo più vicino al nostro modo di “vedere” i colori è il modello HSV





Display

Display raster

- Vari tipi.
  - Display LCD/plasma
  - Display CRT
  - Schermi grandi a proiezione, ecc.
  - Sistemi immersivi, e.g. CAVE, occhiali stereoscopici

Caratteristiche del dispositivo

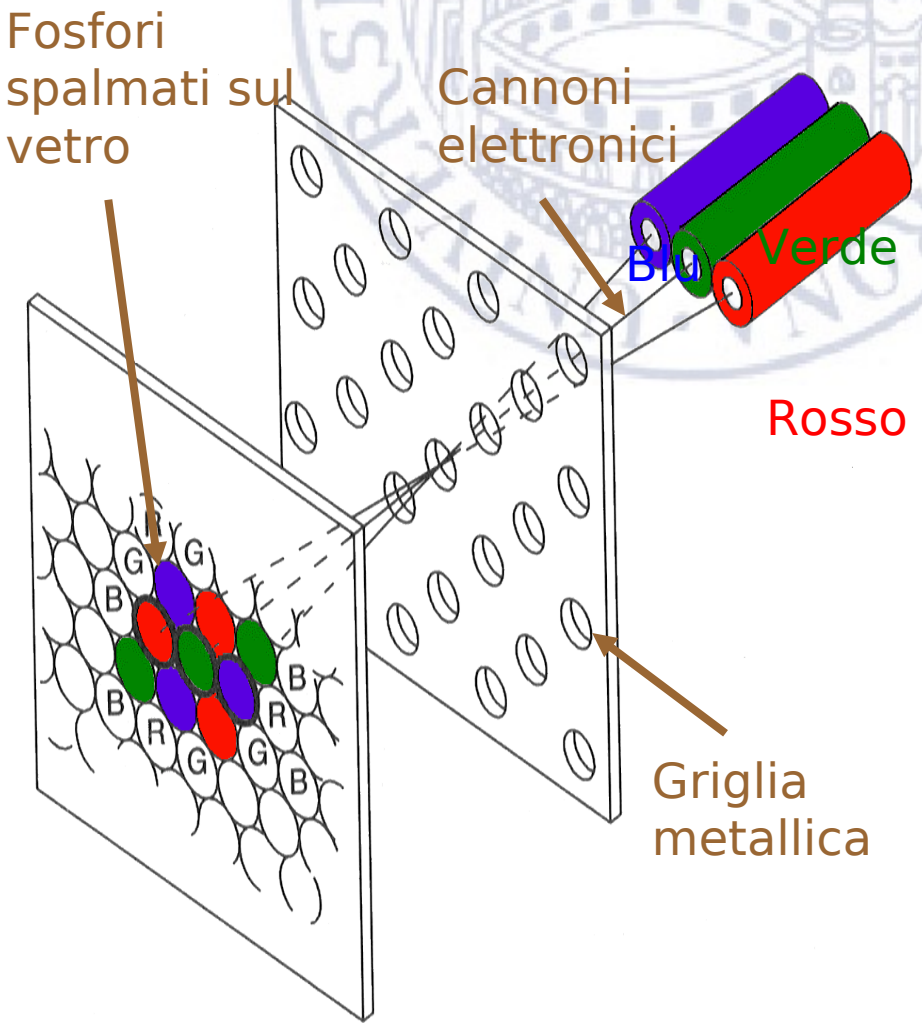
- Dimensione
  - Naturalmente la dimensione fisica è importante per variare l'esperienza di un gioco, ma occorre valutare usabilità per le varie condizioni di uso
- Dot pitch
  - Dimensione pixel
- Aspect ratio
- Refresh rate
- Angolo di visione
- glossy/matte surface
- Contrasto, Nero, riproduzione del colore
  - Si possono fare test percettivi per misurare qualità

Caratteristiche del dispositivo

- Componente principale del sottosistema di pilotaggio del display: i banchi di memoria dedicati alla gestione del display stesso
  - Si chiama memoria di quadro (frame buffer) e contiene le informazioni utili a generare ogni singolo pixel
- La memoria display è di tipo dual-ported, ossia supporta sia la scrittura che la lettura in modo indipendente
  - Lettura con frequenza costante (ordine di 60-85 Hz)
  - Scrittura variabile a seconda dell'applicazione con dati che provengono dal raster subsystem

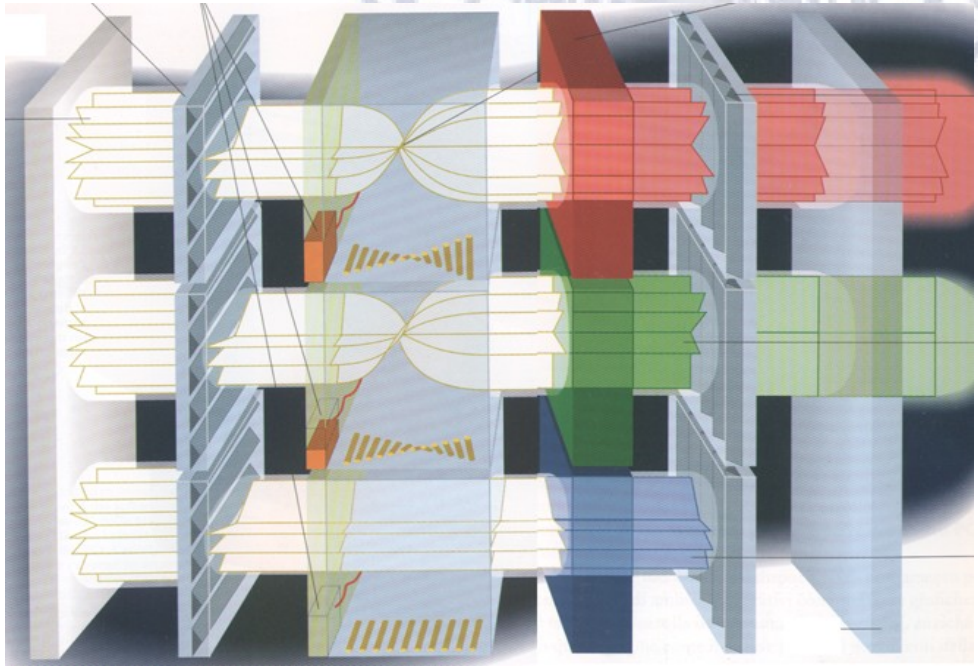
Display CRT

- Il singolo pixel è generato da tre fosfori che emettono luminosità sulle tre bande di colore RGB, di intensità proporzionale a quanto i singoli fosfori siano stati stimolati dal pennello di raggi catodici nella fase di refresh



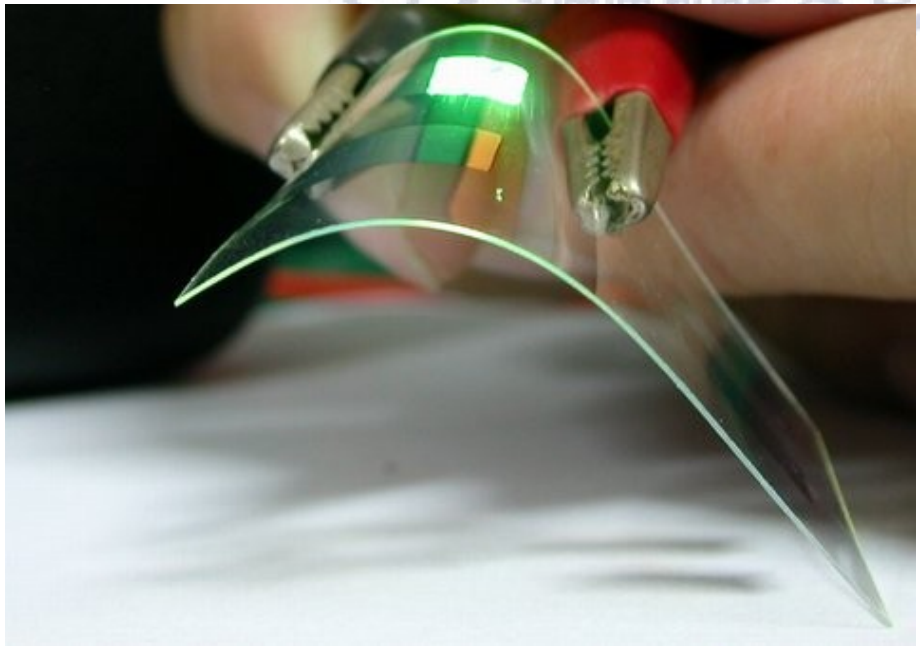
Display TFT

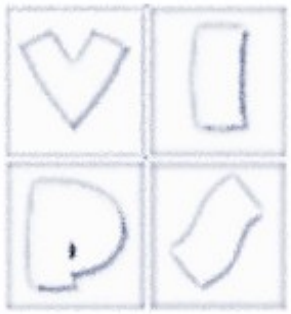
- Piano fluorescente posto al fondo dello schermo
- Nella matrice attiva uno strato di cristalli liquidi guidati da un array di triplette di transistor che “torcono” i cristalli
- Colore tramite filtri colorati



Display OLED

Sottile strato di materiale organico che, in corrispondenza del passaggio di una corrente elettrica si illumina in maniera puntuale
Gli elettrodi sono le due lastre all'interno delle quali è piazzato lo strato (almeno una delle due è trasparente)





Sistemi a proiezione

- Ampia dimensione della superficie visibile
- Basati su tecnologia LCD o DLP (Digital Light Processing)
- LCD: come schermi TFT con proiezione a distanza
- DLP: costituiti da un pannello di micro-specchietti, uno per ogni pixel, di cui si comanda la rotazione su un asse; riflettono la luce incidente in proporzione al loro orientamento; si ottiene maggiore luminosità e nitidezza delle immagini

Sistemi a proiezione

- Problema: risoluzione limitata
- Soluzione: sistemi multi-proiettore (video wall) con suddivisione regolare dell'area di proiezione

Visualizzazione per Realtà Virtuale (immersiva e non)

- VR "non immersiva" == grafica 3D al calcolatore...
  - schermo standard, controllo da tastiera o mouse
  - prospettiva e movimento danno effetto 3D
- VR immersiva
  - visione stereoscopica
  - caschi VR
  - schermo più occhiali oscurati ecc.
  - Tute, guanti, ecc.

Display stereoscopici

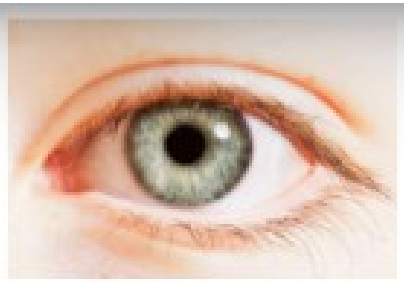
- Principio:
  - fornire due immagini leggermente diverse ai due occhi, in modo da ottenere la percezione di profondità (stereopsi)
- Stereoscopio (fine 19o secolo)
- Anaglifo
  - contiene due immagini sovrapposte, che rappresentano il punto di vista dei due occhi
  - occhiali con filtri cromatici

Head Mounted Display (HMD)

- 2 display LCD, uno per occhio
- LCD shutter glasses
  - Otturatore a LCD che diventa trasparente in sincrono con il display
  - Destra e sinistra alternati rapidamente
  - LCD con filter arrays
  - Matrice di prismi davanti al LCD
  - Pixel pari sull'occhio destro, pixel dispari sull'occhio sinistro
  - L'osservatore deve essere in una zona fissa

Virtual==real?

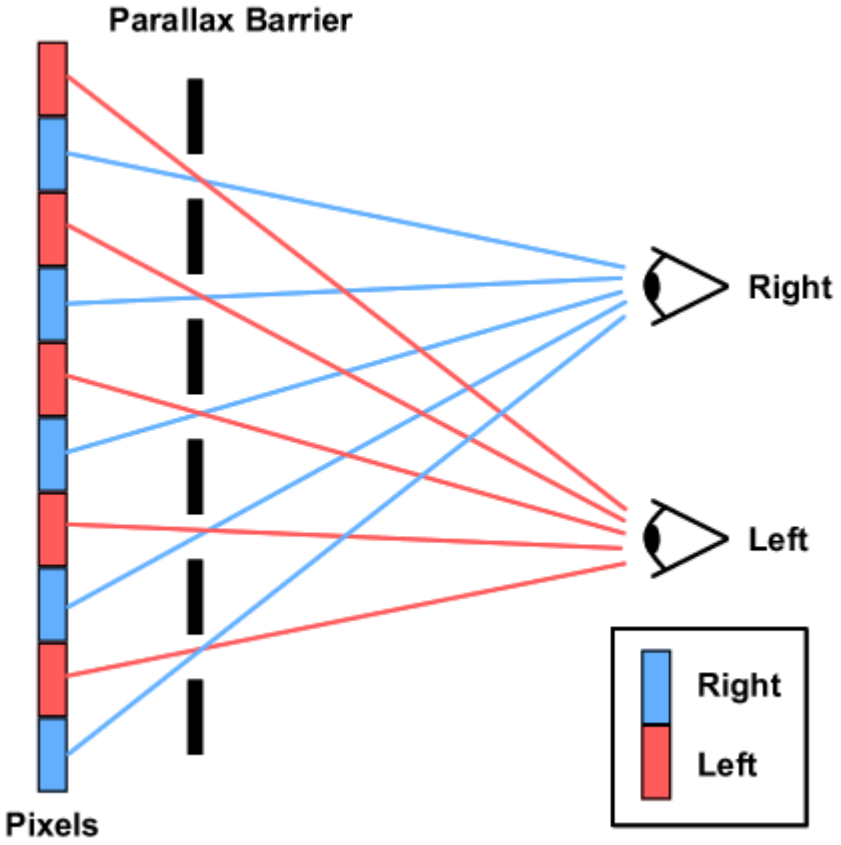
- Not exactly...



	Human Eyes	HTC Vive
FOV	200° x 135°	110° x 110°
Stereo Overlap	120°	110°
Resolution	30,000 x 20,000	2,160 x 1,200
Pixels/inch	>2190 (100mm to screen)	456
Update	60 Hz	90 Hz

Evoluzioni

- Autostereoscopici
  - Sfruttano barriere per far vedere immagini differenti ai due occhi
  - Stanno diventando comuni (es. Nintendo 3DS)
  - Posizione/i fisse



Immersività e realtà virtuale

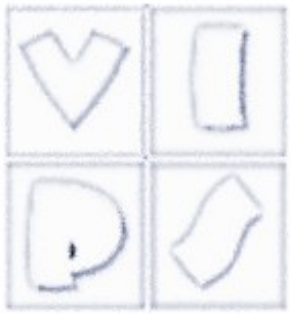
- Naturalmente per sentirsi immersi in una scena 3D occorrerebbe che cambiando il punto di vista cambi la scena opportunamente.
- Coll'autostereoscopico addirittura si perde l'effetto 3D
- Con lo stereo si ha distorsione
- Soluzioni:
  - Tracking della posizione e generazione di una nuova scena (va bene per un utente, complesso
  - Monitor a parallasse continua (c'è comunque distorsione)

Display a parallasse continua

- Es holografika, in pratica molte direzioni con variazione continua

Display e applicazioni

- Le applicazioni di cui ci occuperemo utilizzaranno questi tipi di display raster
- Nei casi di uso di stereo o parallasse continua, si tratta solo di moltiplicare l'output per il numero di immagini da generare
- Maggiore risoluzione accresce complessità
- Display stereoscopici richiedono 2 telecamere virtuali
- Display a parallasse continua ne richiedono molti
  - Array di schede grafiche per il controllo

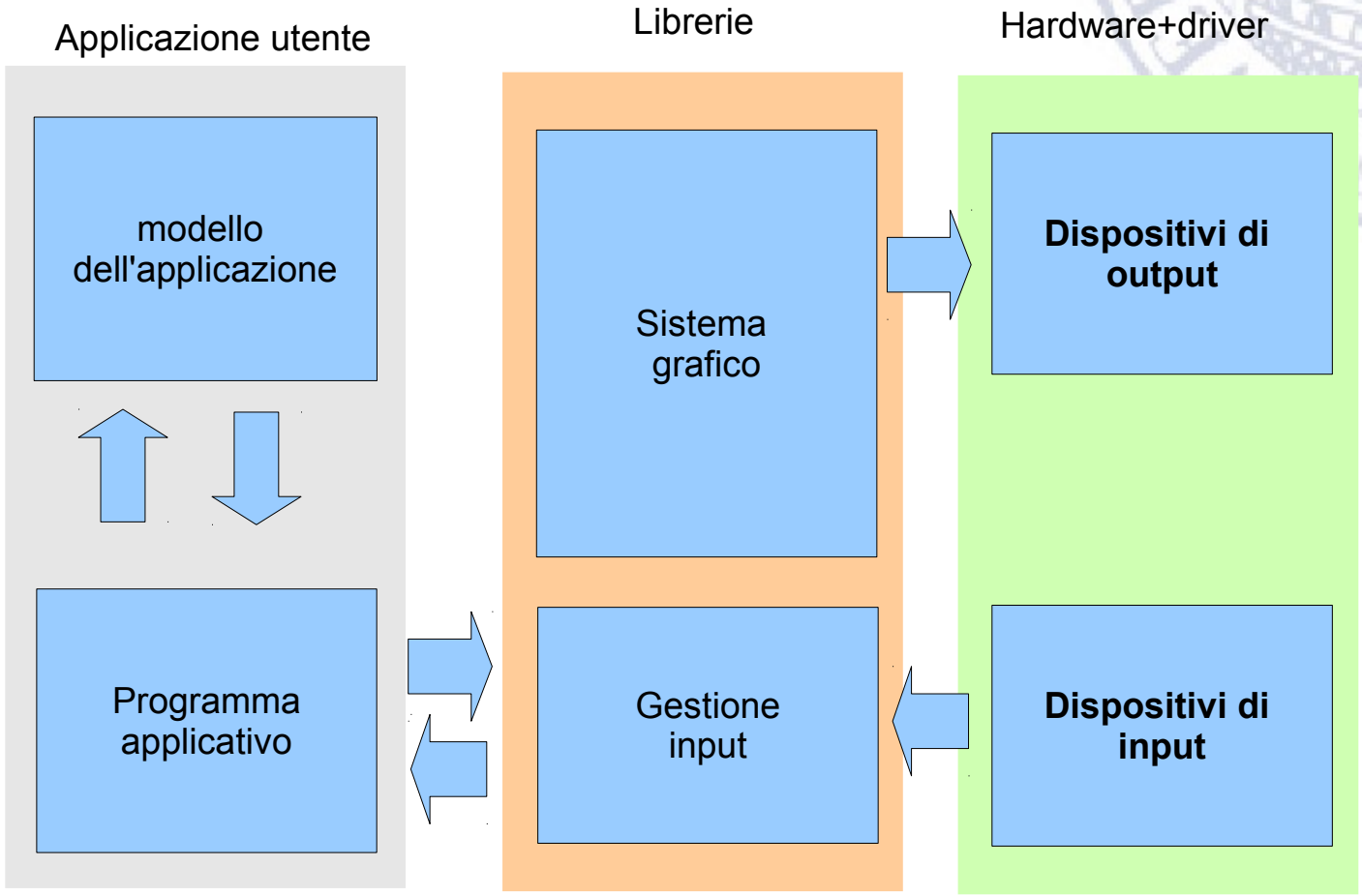


Generazione delle immagini
Il rendering 3D

Schema di un'applicazione grafica

- Vi è una descrizione di qualche tipo (procedurale o meno) del mondo che deve essere rappresentato. La produzione di tale descrizione (modello) prende il nome di modellazione.
- Da tale descrizione si ottiene una immagine visualizzabile da un display tale processo è chiamato globalmente rendering
- La sequenza di procedure ed algoritmi che implementano il rendering prende il nome di pipeline grafica; la studieremo nel dettaglio nel seguito
- Se l'applicazione è interattiva, il disegno dev'essere riprodotto in real time mentre l'utente interagisce con la scena mediante dei dispositivi
- In un gioco occorre gestire naturalmente animazione, fisica, ecc, all'interno dell'applicazione

Schema di un'applicazione grafica

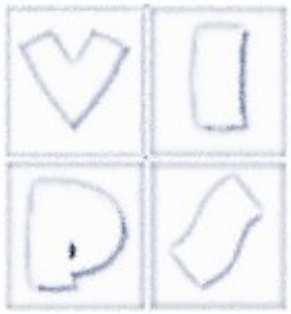


Il modello della scena

- Nelle applicazioni 2D può essere un disegno da riprodurre sul display a meno di una trasformazione geometrica e mappatura sui pixel
- Nelle applicazioni 3D di cui ci interesseremo sarà invece un vero e proprio modello del "mondo" che vogliamo vedere (e con cui vogliamo interagire) e l'immagine sarà generata simulando il processo di acquisizione di immagini di una telecamera "virtuale"
  - Dati gli oggetti della scena, quindi dovremo "simulare" la geometria e la fisica della formazione delle immagini (luce, colore)

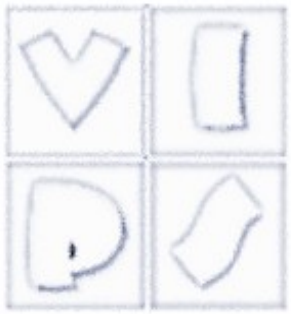
Rendering della scena

- Il passaggio dalla rappresentazione all'immagine si definisce "rendering"
- Comprende tutti gli algoritmi per creare l'immagine per il display, che supporremo voglia immagine raster
- Quindi se partiamo da una rappresentazione 2D (grafica vettoriale) consiste in
  - Trasformazione delle primitive in rappresentazioni di colore sui pixel (rasterizzazione)
  - Eventuale modifica interattiva del disegno
- Se partiamo da una rappresentazione di scena 3D consiste in
  - Proiezione della scena sul piano immagine della telecamera virtuale
  - Trasformazione della scena proiettata in rappresentazioni di colore sui pixel (rasterizzazione)
  - Eventuale interazione con la scena e conseguenteupdate del rendering



Rendering della scena

- Molti calcoli da effettuare
- Complessità dipendente dalla applicazione di interesse:
  - Applicazioni interattive, real-time:
    - Frame rate alto (>10 fps)
    - Tempo di rendering del singolo frame prefissato
    - Si può/deve sacrificare la qualità per garantire l'interattività
  - Applicazioni non interattive (computer animation, grafica pubblicitaria)
    - l'obiettivo primario: massima qualità delle immagini di sintesi
    - non si hanno vincoli sul tempo di generazione del singolo frame
    - animazioni calcolate frame by frame da PC cluster, ricomposte successivamente nella successione temporale corretta

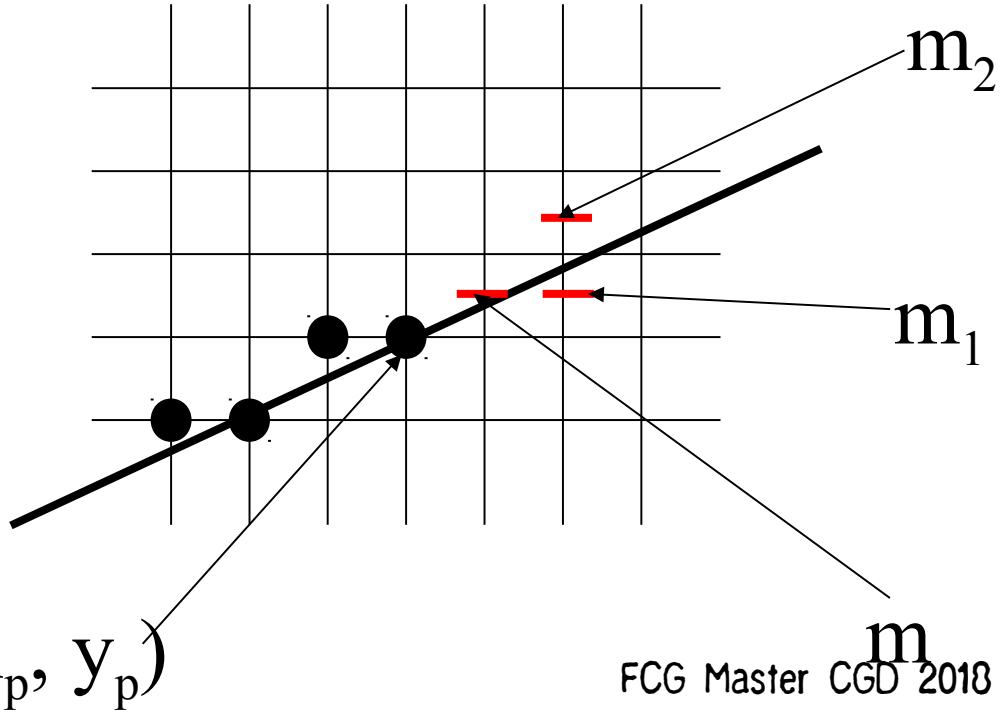


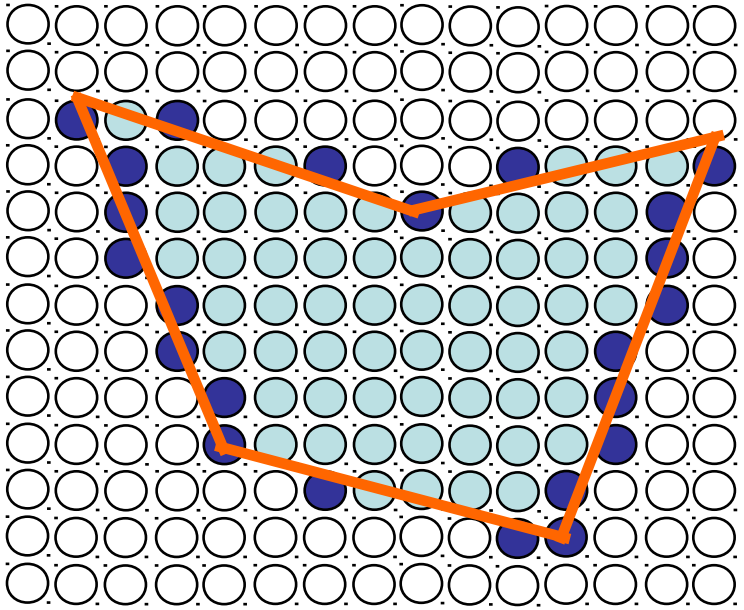
Rendering della scena

- Come si implementa la fase di rendering?
- Applicazioni interattive:
  - si avvalgono pesantemente delle moderne schede grafiche (HW dedicato al processing di dati 3D)
- Applicazioni non interattive:
  - fanno uso di ambienti di rendering più sofisticati e flessibili (ad es. RenderMan), spesso eseguiti SW su cluster di PC

Da disegno a immagine raster

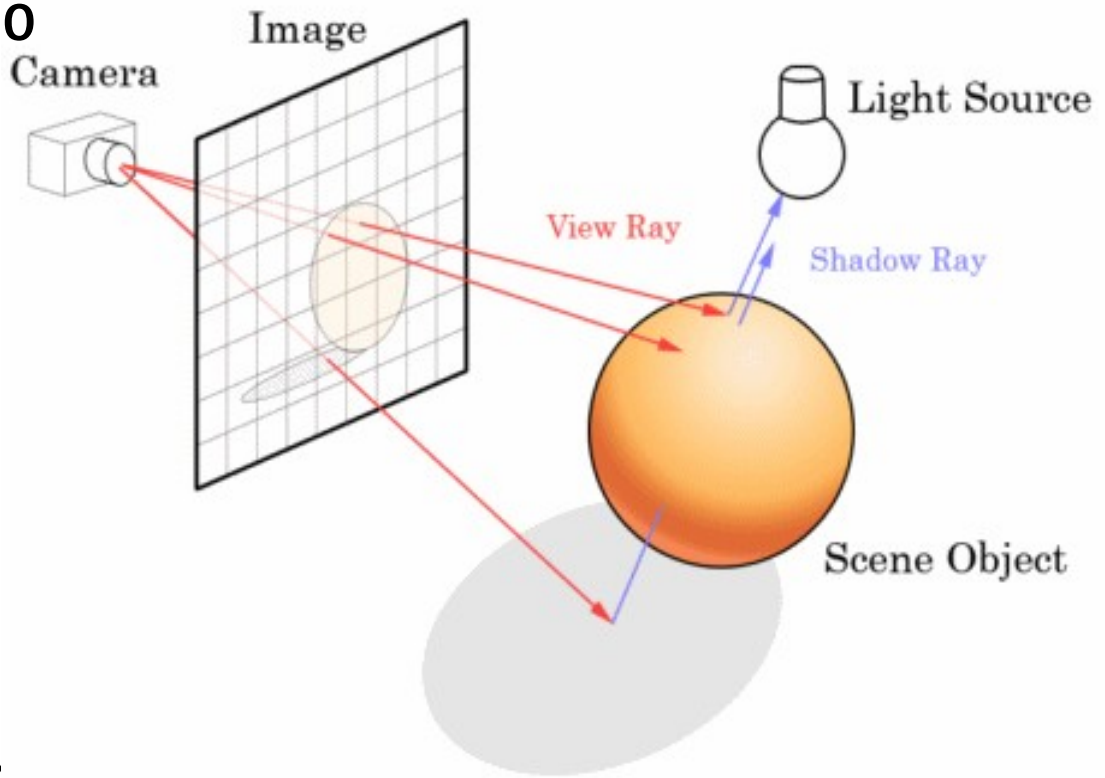
- Per rasterizzare e creare un'immagine da un disegno dobbiamo
  - mappare il disegno (vettoriale: rette, poligoni, cerchi, ecc.) sulla griglia di pixel
  - Applicare algoritmi di rasterizzazione, o scan-conversion
- e.g.
  - Bresenham per rasterizzare segmenti
  - Fill di poligoni





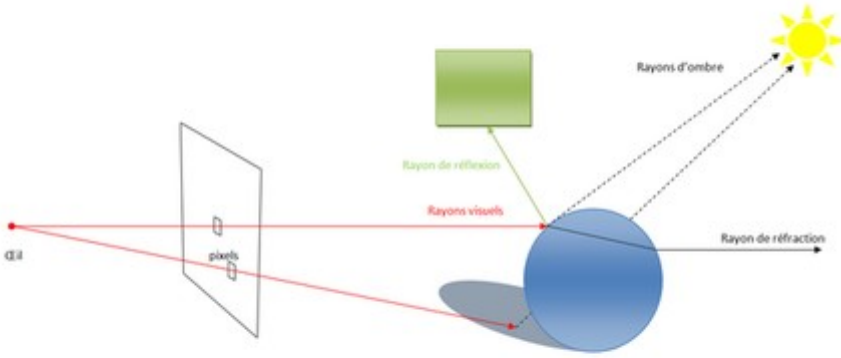
In 3D

- Nella grafica 3D le cose si complicano
  - Modello scena 3D (primitive che descrivono l'oggetto)
  - Modello telecamera (proiezione prospettica)
  - Idealmente posso fare “ray casting”
- Inoltre in generale vorremmo simulare il colore di una scena reale (fotorealismo) quindi
  - Non mappare un colore dell'oggetto
  - Simulare il colore generato dall'illuminazione
  - Problema: fisica complessa



Ray casting/ray tracing

- In effetti andare a calcolare il colore nel punto di intersezione implicherebbe seguire all'indietro il raggio luminoso verso la sorgente
  - Ma la luce arriva al punto da molte direzioni...
- Potrei semplicemente calcolare il colore supponendo che la luce arrivi al punto da una sola sorgente e usando un algoritmo che stimi la luce riflessa (ray tracing base o ray casting)
- Un migliore risultato lo avrei considerando che parte della luce viene riflessa, parte rifratta e posso andare iterativamente a tracciare questi raggi calcolando la componente di luce che incontrano



Fotorealismo

- Uno dei principali scopi della grafica al calcolatore sta nel creare algoritmi per creare dai modelli di oggetti reali immagini che sembrano le foto degli oggetti reali. Per ottenere il fotorealismo occorre:
  - Simulare numericamente l'interazione luce materia e la formazione dell'immagine
  - Questo non è sempre possibile per la complessità
  - In alciuni casi si possono usare “trucchi” per “dare l'impressione” realistica. Nella grafica interattiva è tipico

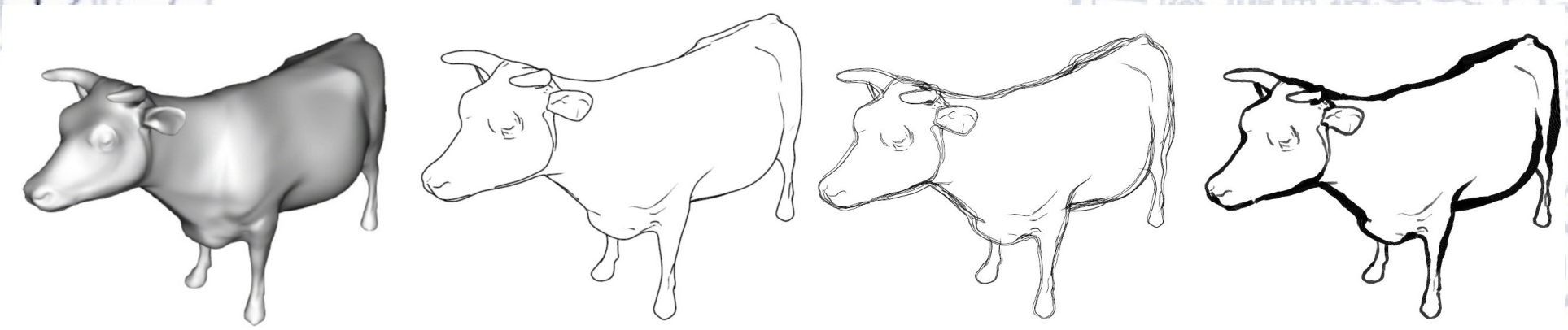
Fotorealismo

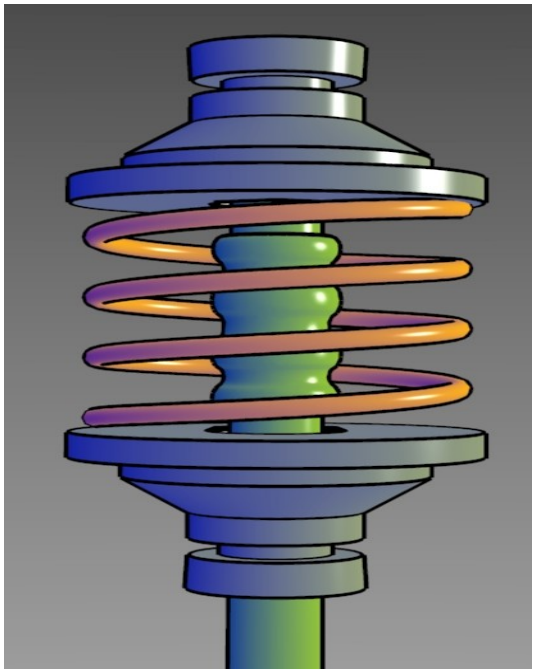
- Richiede come vedremo alogritmi complessi
- Ed anche modelli complessi

Non sempre si cerca il fotorealismo

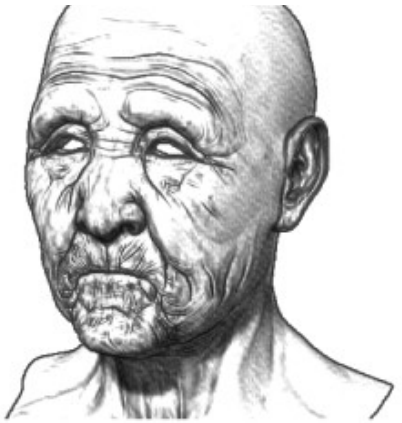
- Non sempre lo scarso fotorealismo è un difetto dovuto a limitate risorse computazionali, interattività, ecc.
- Può essere utile evidenziare contorni, silouhette per il disegno tecnico, ad esempio
- O simulare tratteggio artistico
- O evitare il confronto con la realtà proponendo caratteristiche fantasiose, come nel cinema di animazione
- Si parla di tecniche NPR (Non-Photorealistic Rendering) per generare automaticamente effetti particolari

Non sempre si cerca il fotorealismo





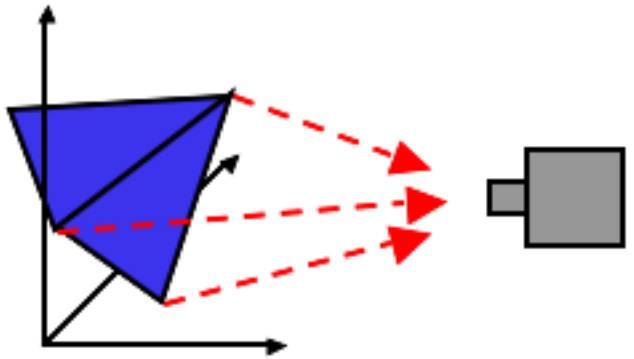
Non-photorealistic = Off



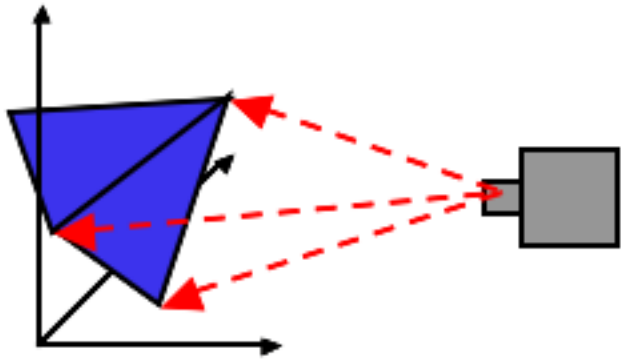
Non-photorealistic = On

Rasterization pipeline

- Invece che gettare (to cast) raggi sulla scena, proietto la scena (composta di poligoni) sul piano immagine.
- La proiezione di un poligono è ancora un poligono, che ha per vertici le proiezioni dei vertici
- La rasterizzazione è un esempio di rendering in object-order, mentre ray-casting è un esempio di rendering in image-order..



Rasterization:
Project geometry forward

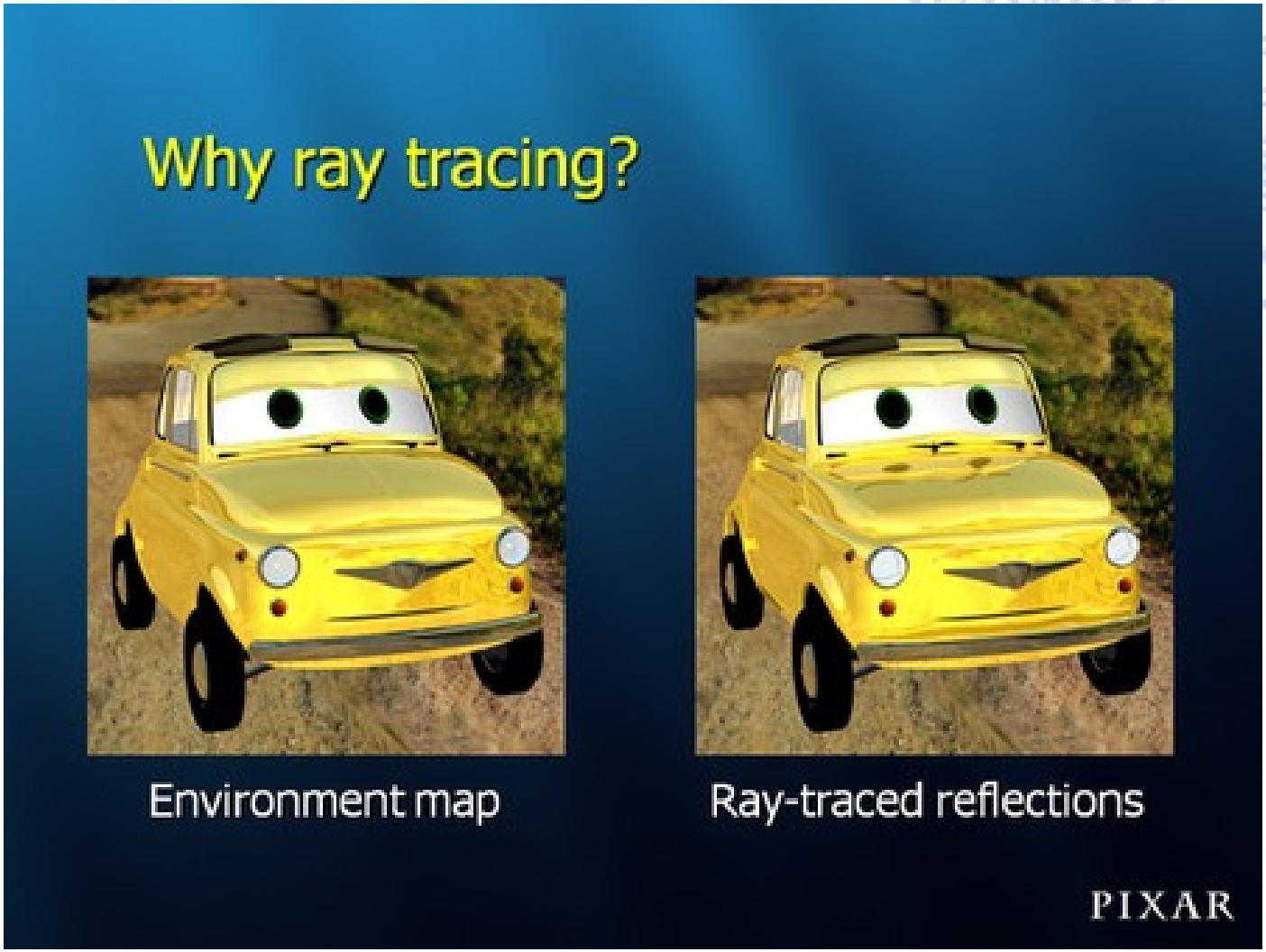


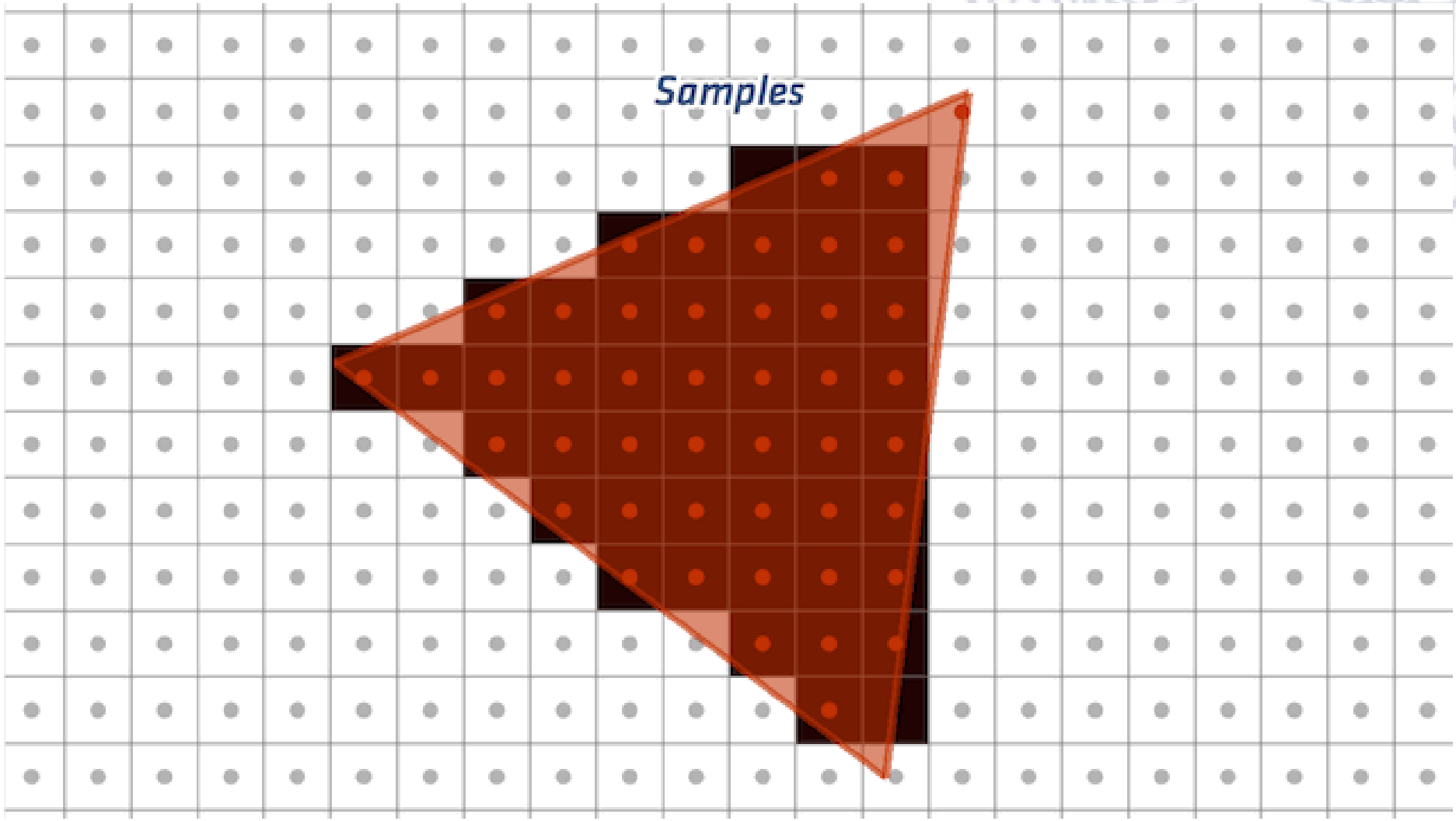
Ray casting:
Project image samples backwards

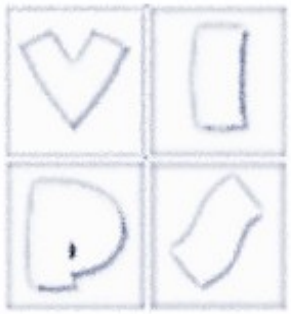
Rasterization vs ray tracing

- Ray tracing
  - Intuitivo (algoritmo del pittore)
  - Indipendente dalla rappresentazione dei modelli
  - Semplice nella sua implementazione base
  - Gestisce naturalmente il calcolo della visibilità delle primitive
  - Complesso calcolare intersezioni
  - Può incorporare il calcolo di trasparenze, ombre
- Rasterization
  - Necessita l'implementazione di vari algoritmi (es gestione visibilità,)
  - Limitato alla rappresentazione poligonale
  - Permette di gestire l'antialiasing
  - Facilmente parallelizzabile
  - Complessità funzione del numero di primitive
  - Storicamente implementato nelle pipeline di accelerazione hardware
  - Non consente di gestire effetti di illuminazione globali senza trucchi, trasparenza, ecc

Rasterization vs ray tracing



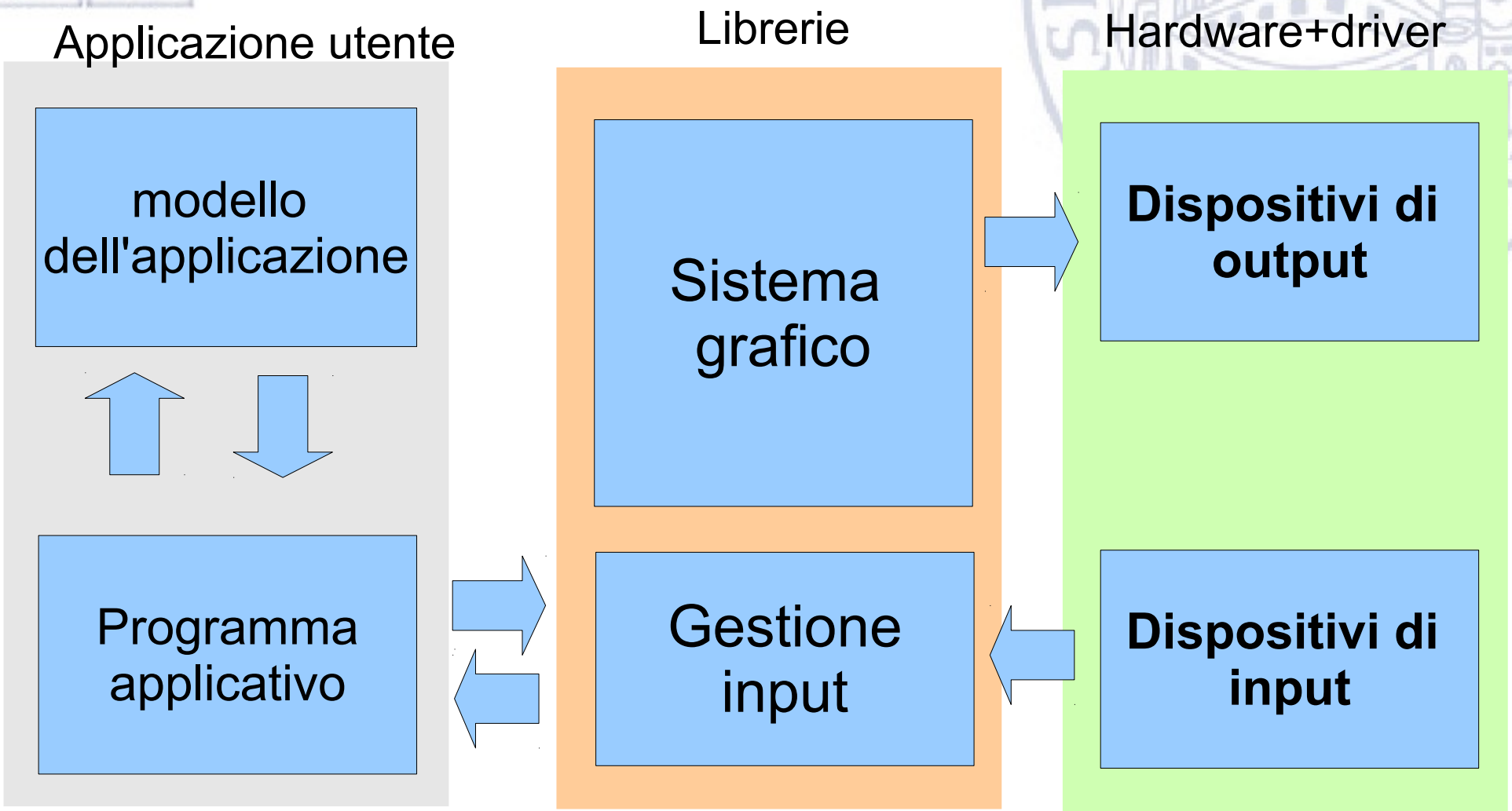

Samples

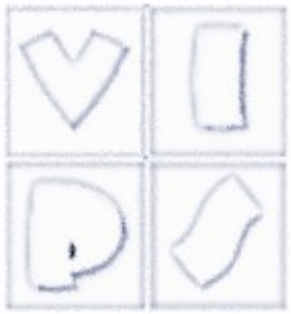


Applicazioni interattive

- I computer con un output grafico interattivo hanno quindi un sistema che gestisce la creazione e l'aggiornamento del disegno sullo schermo
- In origine semplicemente la gestione del frame buffer
- Poi si sono evoluti gli algoritmi di grafica 2D e 3D computazionalmente pesanti e si sono sviluppate architetture per rendere efficiente e disaccoppiare la parte grafica dall'applicazione
- L'applicazione gestisce la “scena” e aggiorna la rappresentazione
- Il sistema grafico la trasforma in immagine

Output e input



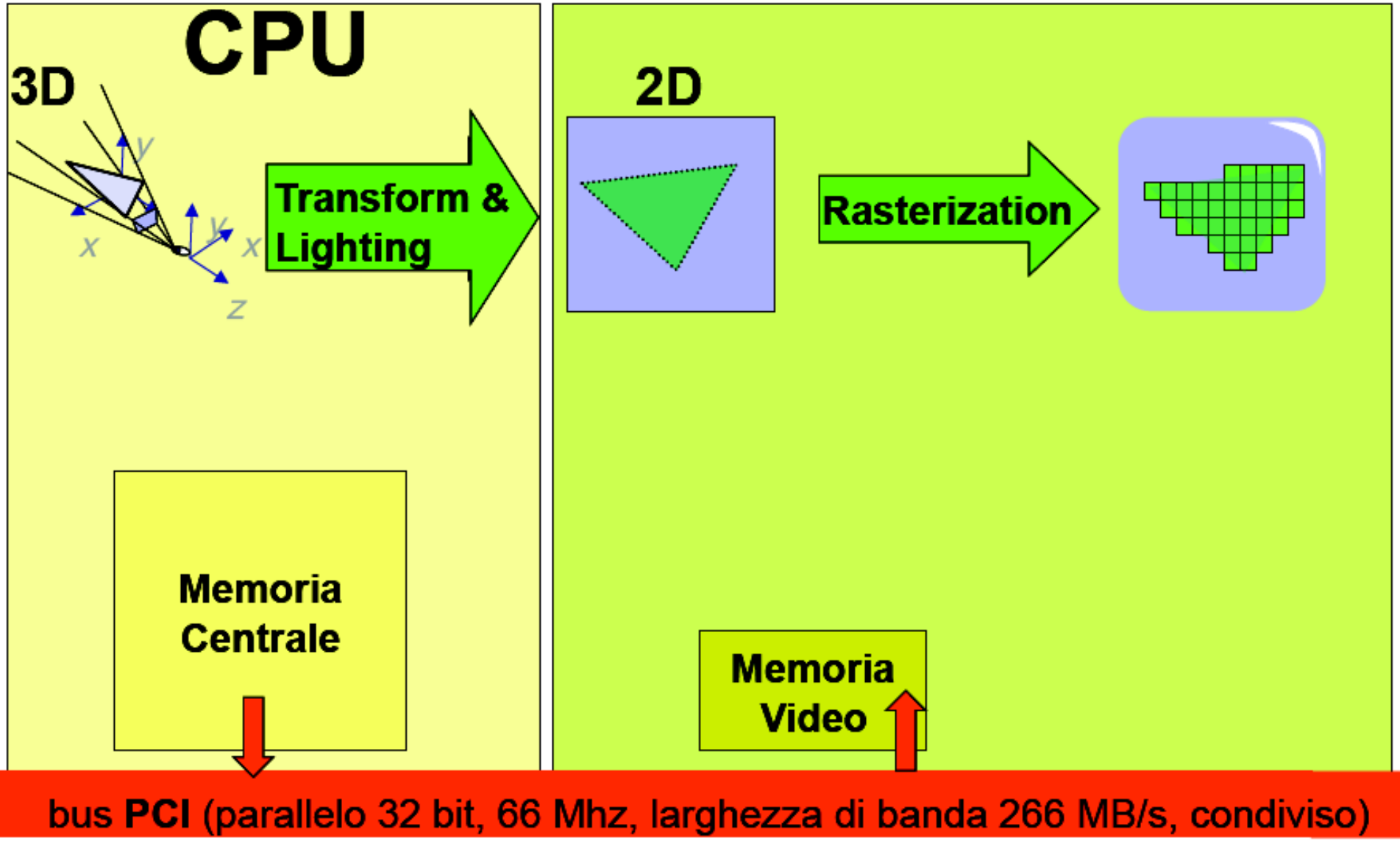


Schede grafiche

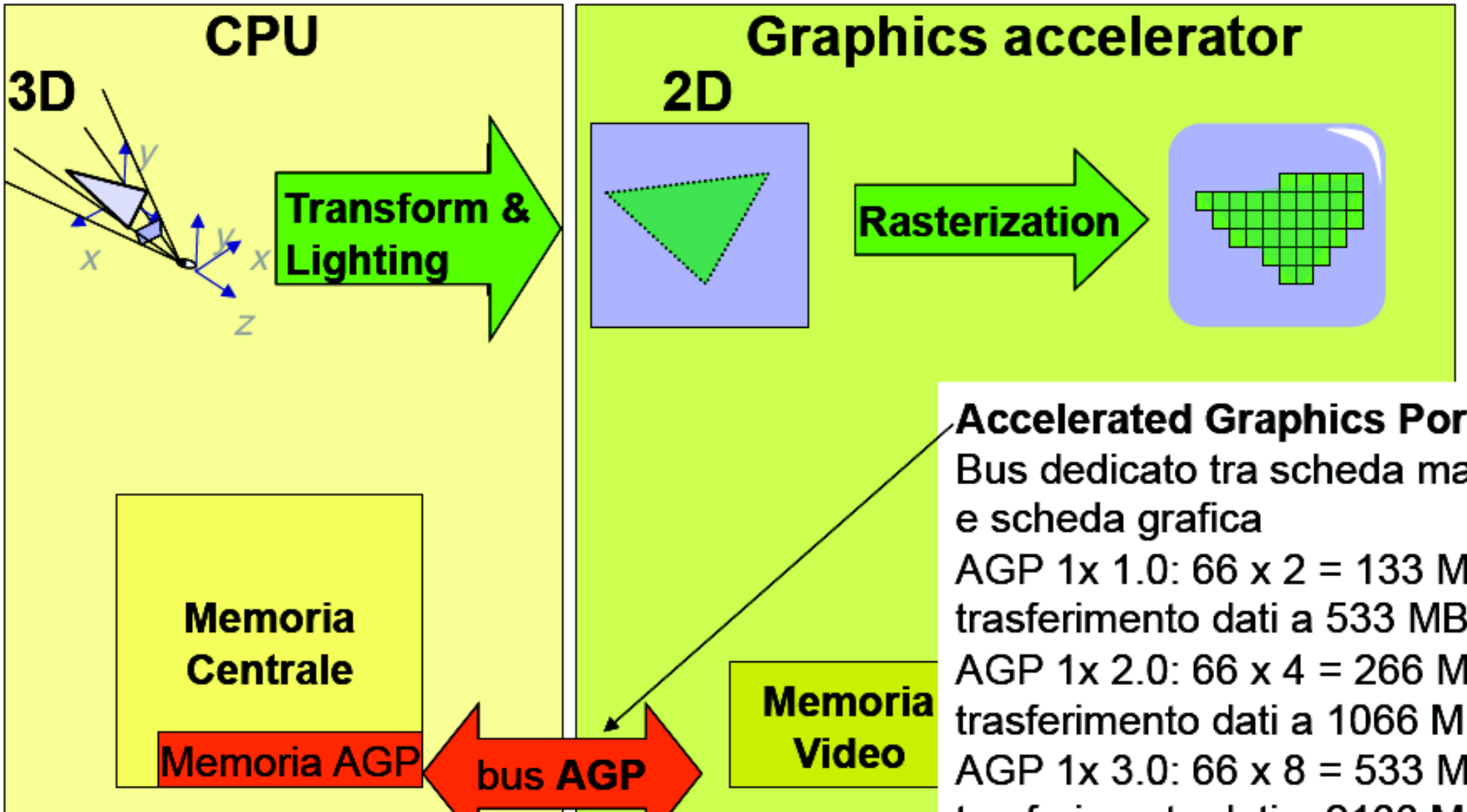
- Gestiscono nei sistemi moderni interattivi tutta la parte di pipeline del sottosistema raster+geometrico
- Non è sempre stato così
- Oggi possono fare molto di più e sono in pratica sistemi di calcolo parallelo
  - Si possono implementare algoritmi di rendering più complessi della pipeline standard
  - Si può fare calcolo generico (GPGPU)

Evoluzione storica

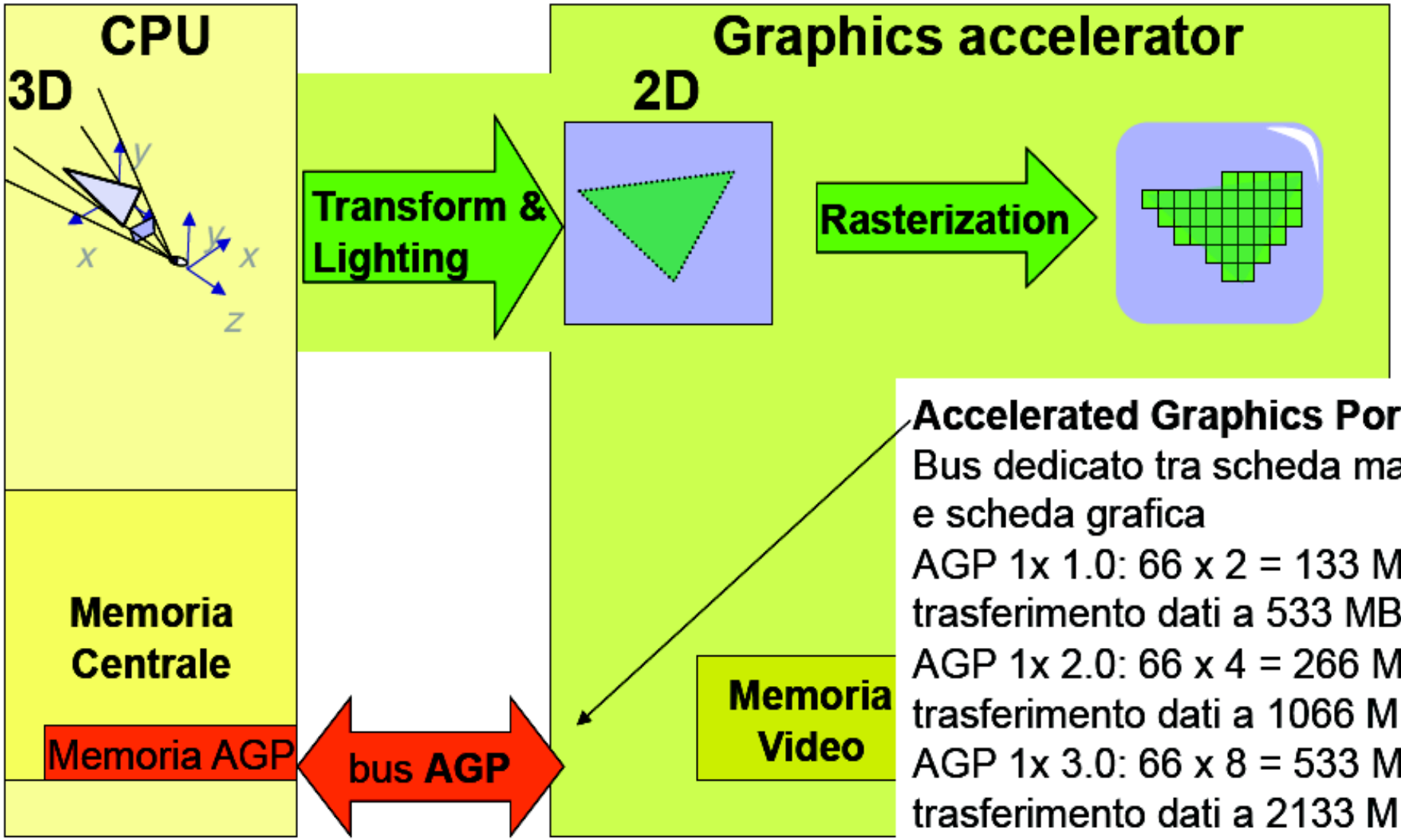
- 1995-1997: 3DFX Voodoo



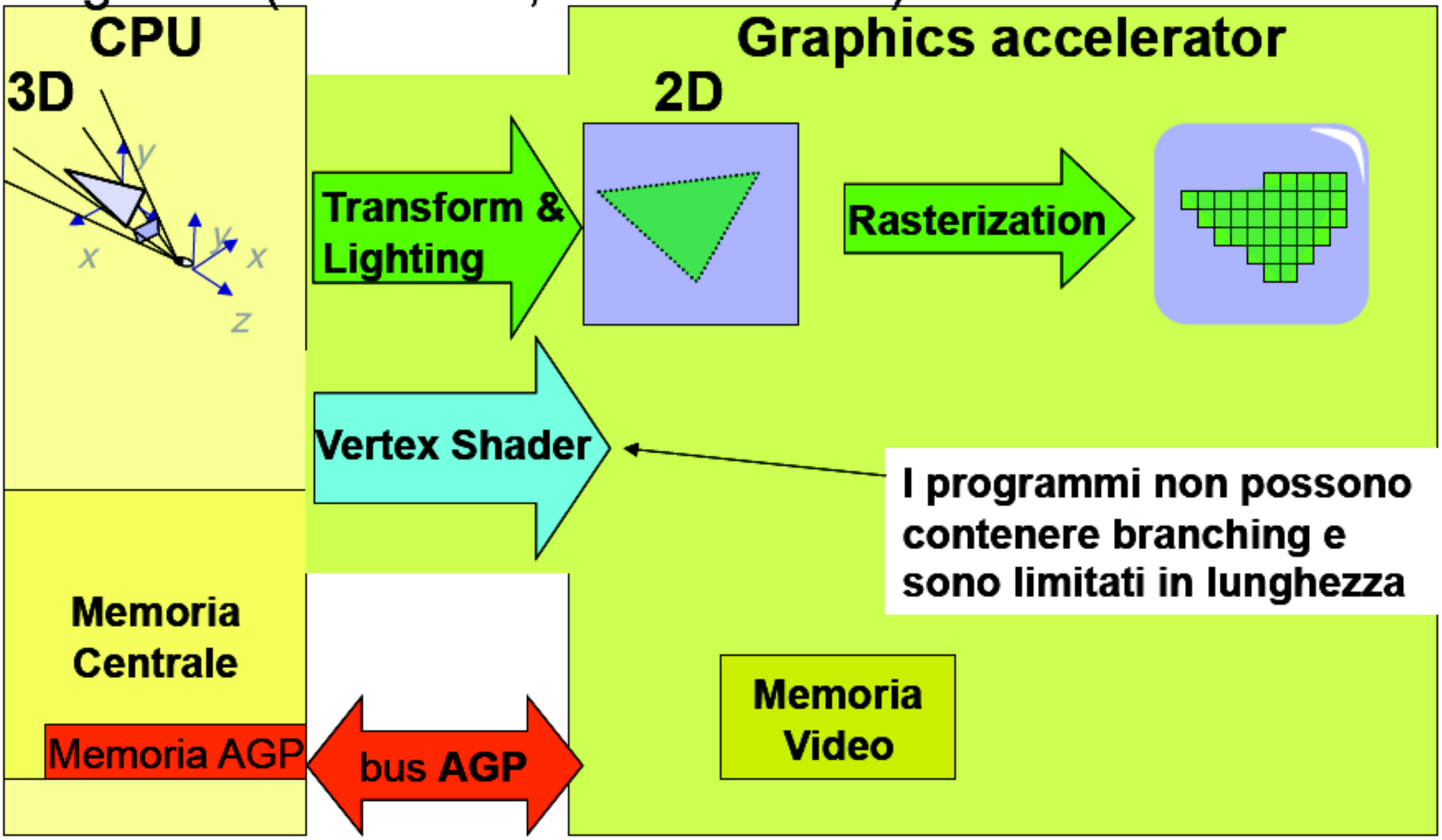
1998: NVidia TNT, ATI Rage



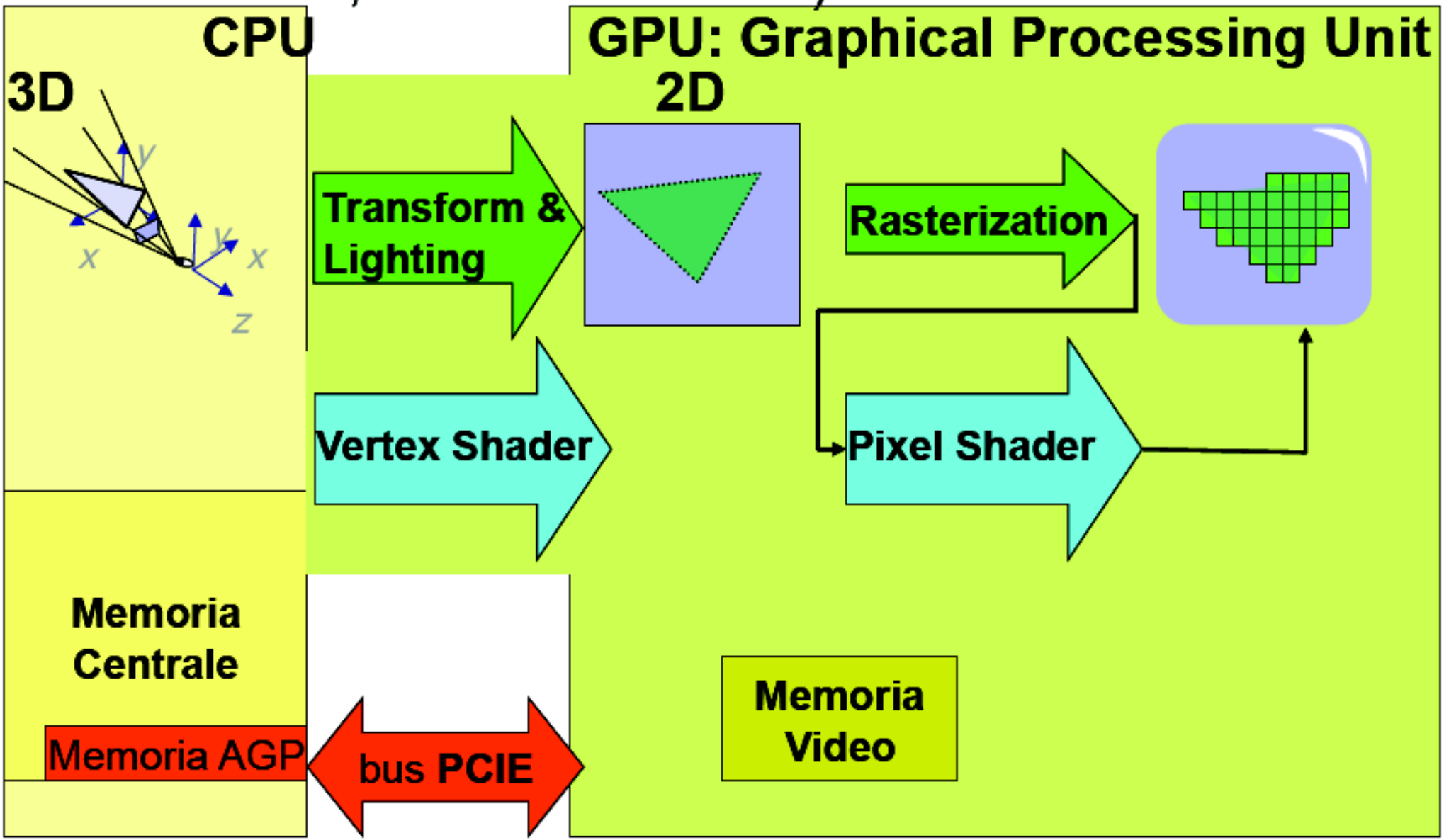
- 1998: NVidia TNT, ATI Rage



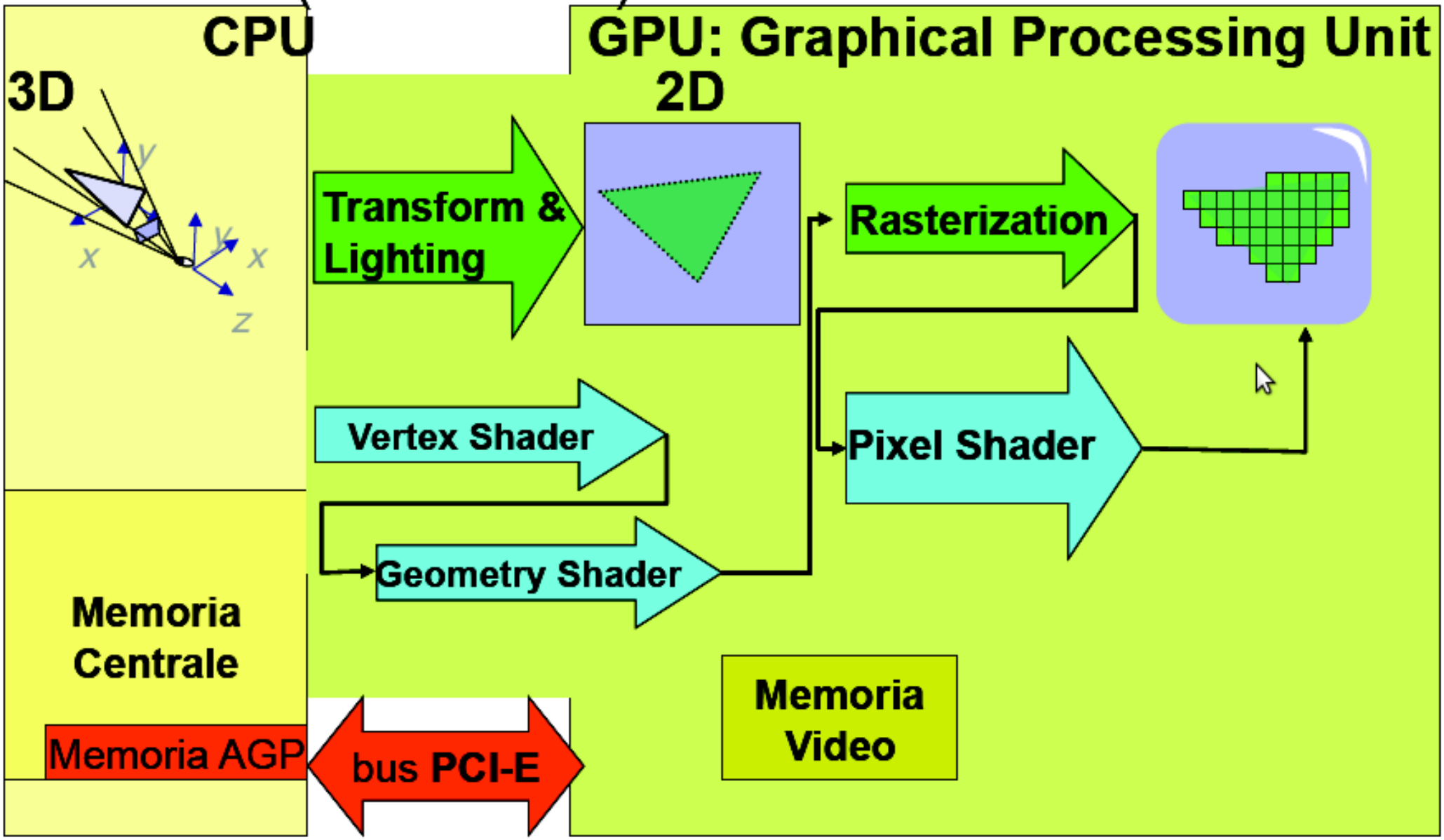
- 2001: Vertex Shader. Programmare il chip della scheda grafica (GeForce3, Radeon 8500)



- 2004-5: PCI-Express, branching negli shader (GeForce 6800-7800, ATI Radeon 9200)



- 2007: geometry shader, stesso hardware per tutti gli shaders (NVidia 8800)



Visualizzazione della scena - API

- API (Application Programming Interface) per la grafica 3D
  - OpenGL, DirectX
- Progettate per grafica 3D interattiva, organizzazione logica funzionale ad una efficiente implementabilità HW
- Efficienza direttamente dipendente dalla possibilità di elaborare in parallelo le diverse fasi del processo di rendering
- Soluzione vincente: suddivisione del processo in fasi indipendenti, organizzabili in pipeline
  - Maggiore parallelismo e velocità di rendering
  - Minore memoria (fasi indipendenti -> memoria locale, non necessario conoscere la rappresentazione dell'intera scena ma solo della porzione trattata)

API per la grafica/storia

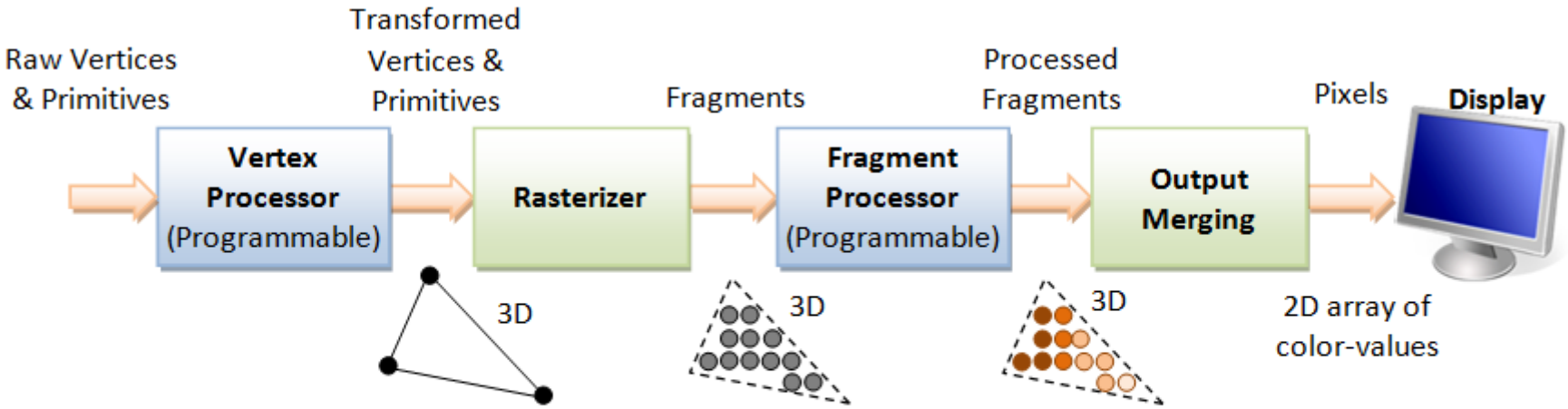
- GKS (Graphics Kernel System) primo standard europeo per grafica 2D. Poi estensione 3D
- PHIGS Programmer Hierarchical Interactive Graphics System appoggiatto dall'ANSI. Separazione modelling/rendering e modello applicazione/programma applicativo
- Intanto Silicon Graphics crea API proprietaria IrisGL direttamente legata all'architettura, efficiente
- OpenGL versione “aperta” di IrisGL (1992)
  - Fornisce un'interfaccia uniforme all'hardware grafico
  - Emulazione software
- Alternativa: Microsoft Direct 3D (DirectX)

Note

- OpenGL si occupa solo del rendering grafico. Altre librerie per le finestre, l'input, funzioni accessorie
  - GLX, X11, GLU, GLUT, GLFW...
- Oggi si usano spesso librerie di livello superiore, per esempio OpenSceneGraph, Java3D, ecc per modellare a oggetti sistemi grafici interattivi
- Concetto di Scene Graph
  - Struttura dati con schema logico e spaziale della scena
- Standard X3D, evoluzione di VRML per la realtà virtuale
- Tool per sviluppo giochi, (game engine), ma non solo, es Unity 3D

Pipeline di rendering interattivo

- Tre principali fasi elaborative:
  - gestione e trasmissione della rappresentazione tridimensionale (a cura dell'applicazione)
  - gestione della geometria (Geometry Subsystem)
  - gestione della rasterizzazione (Raster Subsystem)



3D Graphics Rendering Pipeline: Output of one stage is fed as input of the next stage. A vertex has attributes such as (x, y, z) position, color (RGB or RGBA), vertex-normal (n_x, n_y, n_z), and texture. A primitive is made up of one or more vertices. The rasterizer raster-scans each primitive to produce a set of grid-aligned fragments, by interpolating the vertices.

Modello dell'applicazione/programma

- Nell'applicazione ci saranno le scene (sistemi di riferimento spaziali) e gli oggetti per generare le imamgini
  - Modelli geometrici 3D, tipicamente mesh di triangoli, con coordinate punti e connettività
  - Alternativa: rappresentazioni volumetriche. Tipicamente cubetti pieni/vuoti (voxel) con caratteristiche dei materiali
  - Molte immagini (noto il punto di vista si può interpolare creando applicazioni grafiche interattive
- Il tutto deriva/è gestito dall'applicazione
  - Giochi
  - Dati da visualizzare (visualizzazione scientifica)
  - Simulazioni, ecc.

Input per la grafica interattiva 3D

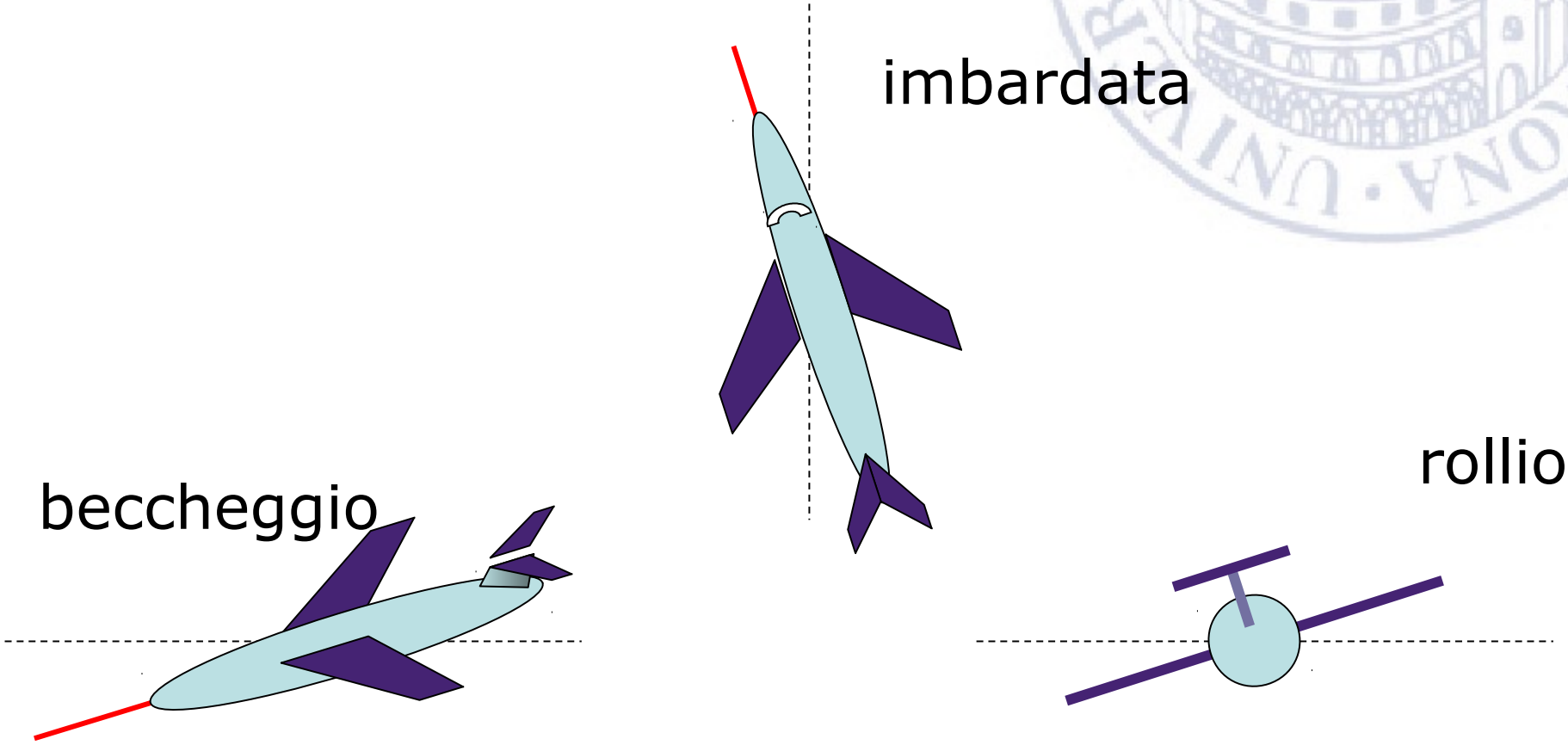
- Dipende dall'applicazione
- Interazione per visualizzazione
  - Selezione, manipolazione
- Per videogiochi
  - Movimento camera, oggetti nella scena

- Non c'è corrispondenza in genere tra movimento e risultato sullo schermo come con il 2D
  - Occorre usare convenzioni
  - O dispositivi di tracking 3D

Device per posizionamento 3D

- mouse 3D
  - Strumento con 6 gradi di libertà (DOF): x, y, z + rollio, beccheggio, imbardata, non diretti
- Simulazione, es. cabina di pilotaggio e controlli virtuali
  - volanti, manopole e rotelle, come nella realtà!
- Accelerometro
- Giroscopio
- Visione stereo
- Sensori attivi con visione IR (es. Microsoft Kinect), telecamere a tempo di volo
- guanto interattivo
  - fibre ottiche per rilevare la posizione delle dita

Orientazione in 3D

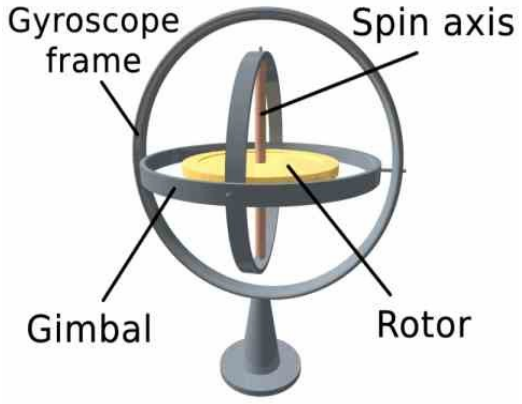


Accelerometro

- Può rilevare l'accelerazione su assi selezionati
- Idea di fondo semplice: pensate a un sistema come quello in figura massa-molla (estensimetro)
- Dato che siamo nel campo gravitazionale, possiamo calcolare l'orientazione di un oggetto in base alla direzione del campo rispetto all'accelerometro (specie se con più assi)
- Problema: non possiamo proprio calcolare l'angolo di imbardata: non cambia infatti il campo gravitazionale sull'oggetto

Giroscopio

- Permette di calcolare anche l'angolo di imbardata
- Il principio di funzionamento è quello della conservazione del momento angolare: se un oggetto ne ha, si oppone al tentativo di cambiare orientamento dell'asse di rotazione
- Anche dei giroscopi esistono versioni miniaturizzate e a basso costo
  - Incluse in smartphone, controlli videogiochi e non solo

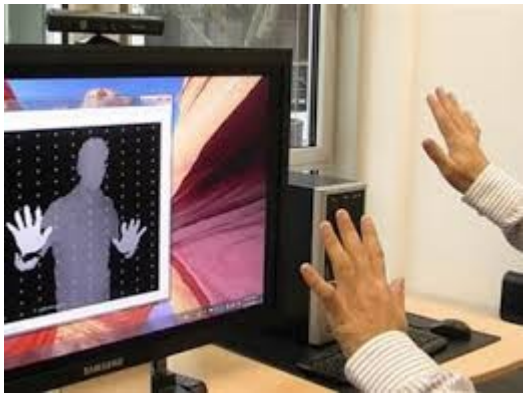


Nintendo WII

- Nintendo WII, controllo 3D con accelerometro e telecamera a IR:
  - Orientamento 3D (parziale)
  - Posizione mediante telecamera IR
  - Barra con led IR a distanza nota, SW che fa detezione di blob e triangolazione
  - Sottoprodotto: si può usare per tracking di penne a IR
  - Esempio: whiteboard
- Wii Plus: aggiunge giroscopio, può fare tracking posizione 3D assoluto

Microsoft Kinect

- Cattura mappa di profondità (oltre che immagini RGB) a frame rate interattivo
- In più acquisisce audio, immagine IR con pattern, ha modulo integrato per riconoscere modelli umani e fare tracking di scheletro.
- Basso costo, drivers e SDK disponibili (Microsoft, OpenNI)



Kinect 2

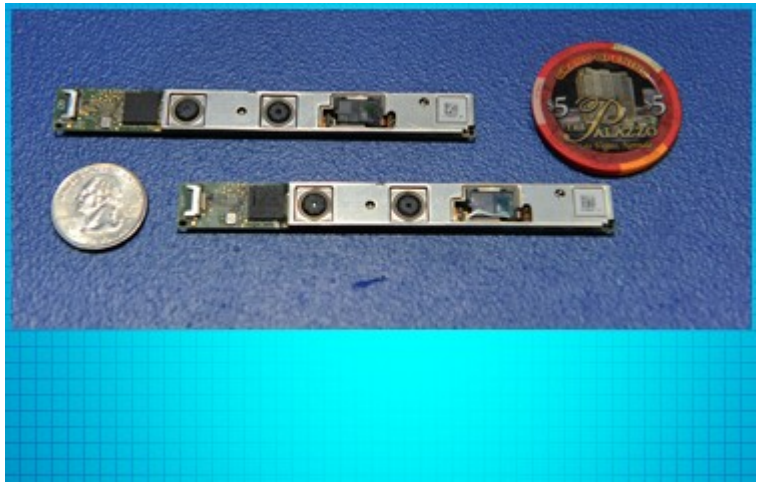
- Contiene una camera a tempo di volo (time of flight)

LeapMotion controller

- Sempre basato su visione 3D
- Sistema stereo con illuminatore a infrarosso
- Il software stima posizione di mani e dita (puntatori)
- Dichara accuratezza nel posizionamento molto elevata
- Latenza bassissima
- Costo basso (100 Euro)

Intel RealSense

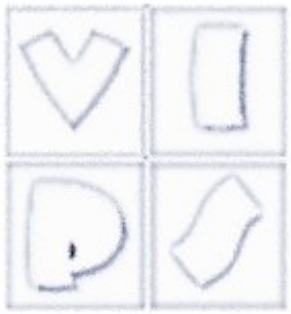
- Altro sensore con tecnologia simile e costo basso
- API per tracking e riconoscimento gesti



VR controllers

Gestione dell'input

- Le API grafiche non si occupano della gestione dell'input
- Gestione dell'input demandata in larga parte al sistema operativo, tramite window manager
- Esempi
  - apertura e chiusura di finestre
  - gestione dei dispositivi di input (mouse, tastiera ecc.)
  - dispositivi di I/O per il canale audio
- Librerie specifiche (GLUT, GLFW, Qt) si possono integrare con quelle grafiche e interfacciano le API con l'ambiente a finestre e di gestione dell'input (e offrono funzionalità più o meno sofisticate)
- Ambiente trasparente al programmatore

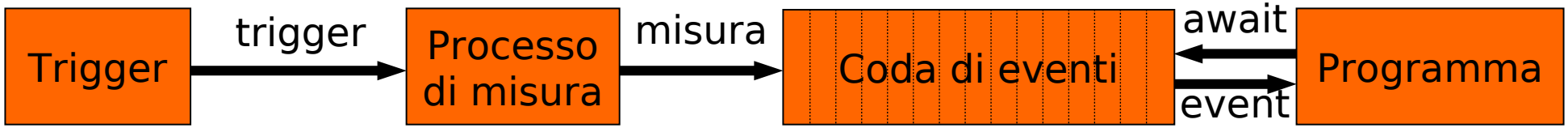


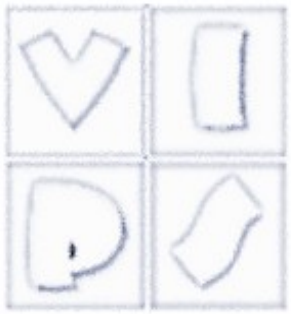
Il modello ad eventi

- Gestione dell'input disaccoppiata dal suo uso da parte del programma applicativo
- L'applicazione opera in un ambiente con molteplici dispositivi di input, ciascuno con un proprio processo di misura e un trigger
- Ogni volta che si attua il trigger di uno di questi dispositivi, viene generato un evento
- La misura, insieme all'identificatore del dispositivo che l'ha eseguita, viene memorizzata in maniera asincrona in una event queue (coda di eventi)

Il modello ad eventi

- Il programma utente può esaminare l'evento in testa alla coda se esiste, oppure può attendere che un evento si verifichi
- Il programma applicativo consuma gli eventi al momento in cui ha disponibile il time-slot per processarli

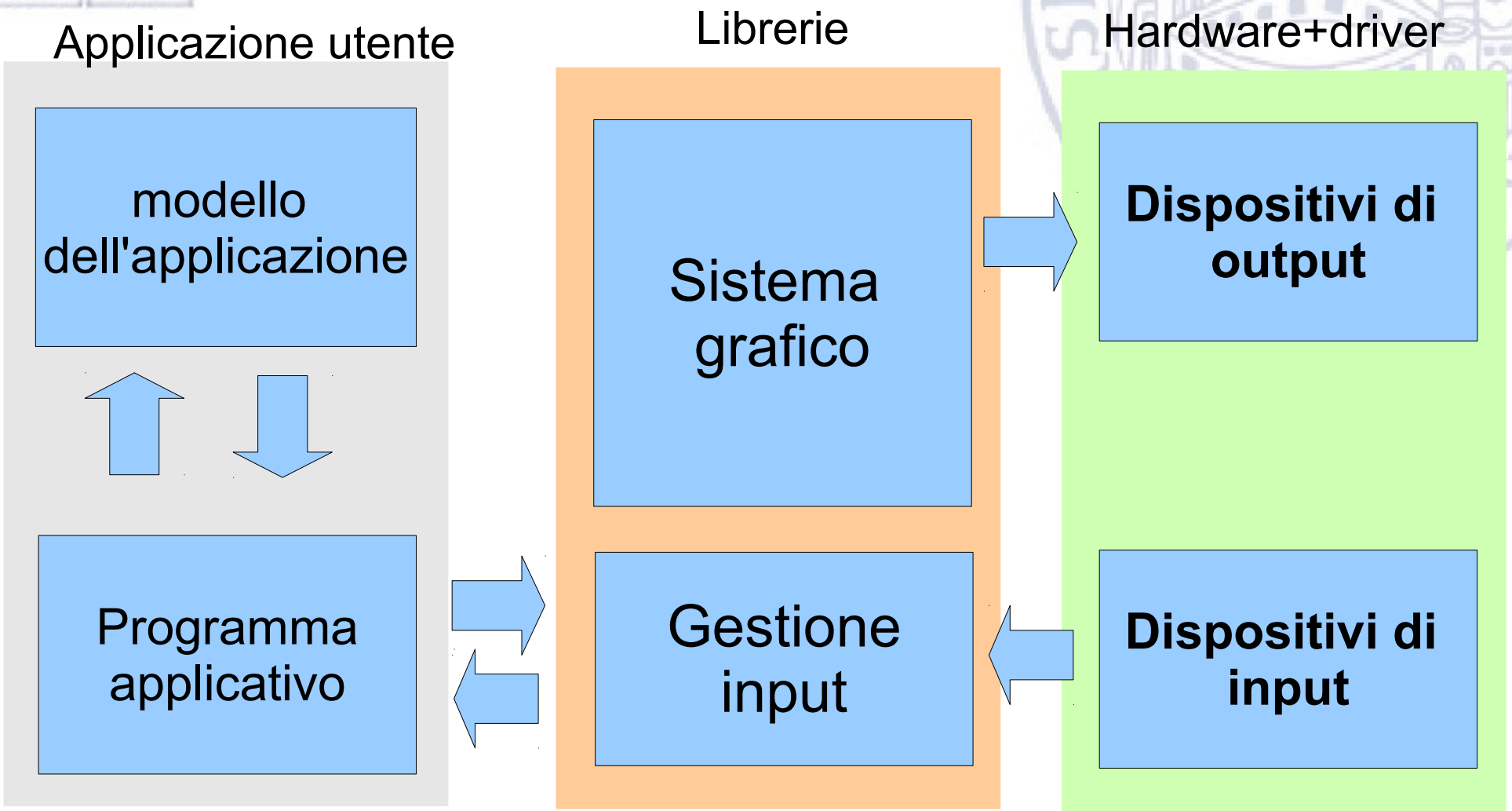


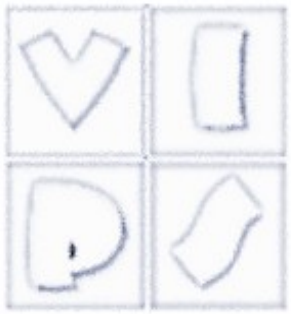


Callback

- Questo schema caratterizza gli ambienti client-server e le applicazioni interattive, in cui più processi concorrenti controllano le varie componenti del sistema
- Un modo comune per gestire dispositivi in questo contesto è quello di fornire una opportuna funzione (callback) per la gestione di ogni specifico tipo di evento
- Tale funzione e attivata dal gestore degli eventi del sistema a finestre senza che vi sia un esplicito controllo di attivazione da parte del programma applicativo
- È molto più semplice progettare e controllare sistemi complessi con interfaccia costituita da molte componenti indipendenti

Output e input





Riferimenti

- Ganovelli et al cap 1
- Scateni et al introduzione
- Angel cap.1