CO3091 - Computational Intelligence and Software Engineering

Lecture 13

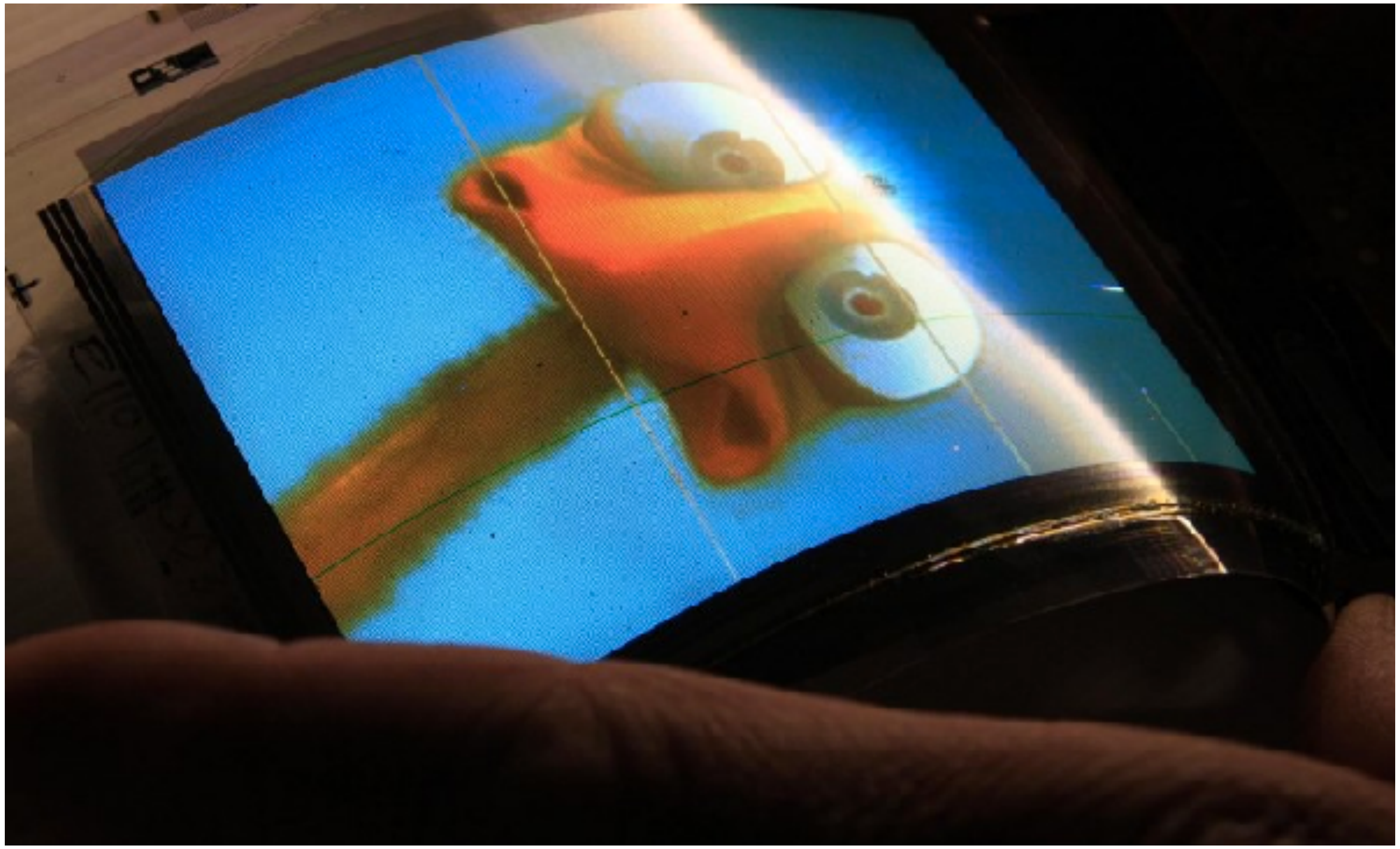


Image from: http://cdn.slashgear.com/wp-content/uploads/2013/04/flexible_oled-820x420.jpg

Software Energy Consumption Optimisation

Leandro L. Minku

Announcements

- Fire drill during lecture 13 on multi-objective evolutionary algorithms.
- Last slides of lecture 13 will be covered in tomorrow's surgery.
- A brief recap of the concepts learnt in lecture 13 will also be given.

Overview

- Mobile apps and the energy consumption of OLED screens.
- Formulate the problem of minimising energy consumption of OLED screens as an optimisation problem.
- A multi-objective evolutionary algorithm design for this problem.

Mobile Apps

- Mobile apps have widespread use.
- Mobile apps consume energy (battery).
- Surveys show that battery consumption is one of the key factors considered by users when choosing a mobile app.
- In order to reduce energy consumption, one can concentrate on how mobile apps use energy-greedy hardware components, e.g., GPS, Wi-Fi, or the screen.

Screens…

Old-Style Cathode-Ray Tubes (CRTs) TVs

Image from: http://robonwriting.com/wp-content/uploads/2013/04/OldTV.jpg

The biggest ones were about 30–60cm (1–2ft) deep and almost too heavy to lift by yourself.

1940s TVs

Image from: http://3.bp.blogspot.com/-vkaOVPMiqk8/VPcydzeDtEI/AAAAAAAABeg/hUutzQmatK8/s1600/TVeurope_TV_Baird_T-18_1938.JPG

Liquid Crystal Display (LCD) TVs

Image from: http://web.tradekorea.com/upload_file2/product/000/P00241000/cbe9caa5_b5a7181c_5fd3_42aa_bee2_1e29eb232e1a.jpg

LED-backlit LCD TV (LED TVs)

Image from: http://cdn.ndtv.com/tech/intex_led_tv_4300_fhd.jpg

Organic LED (OLED) TVs

Image from: http://icdn2.digitaltrends.com/image/lg-65ec9700-lined-up-970x647-c.jpg

OLED Mobiles

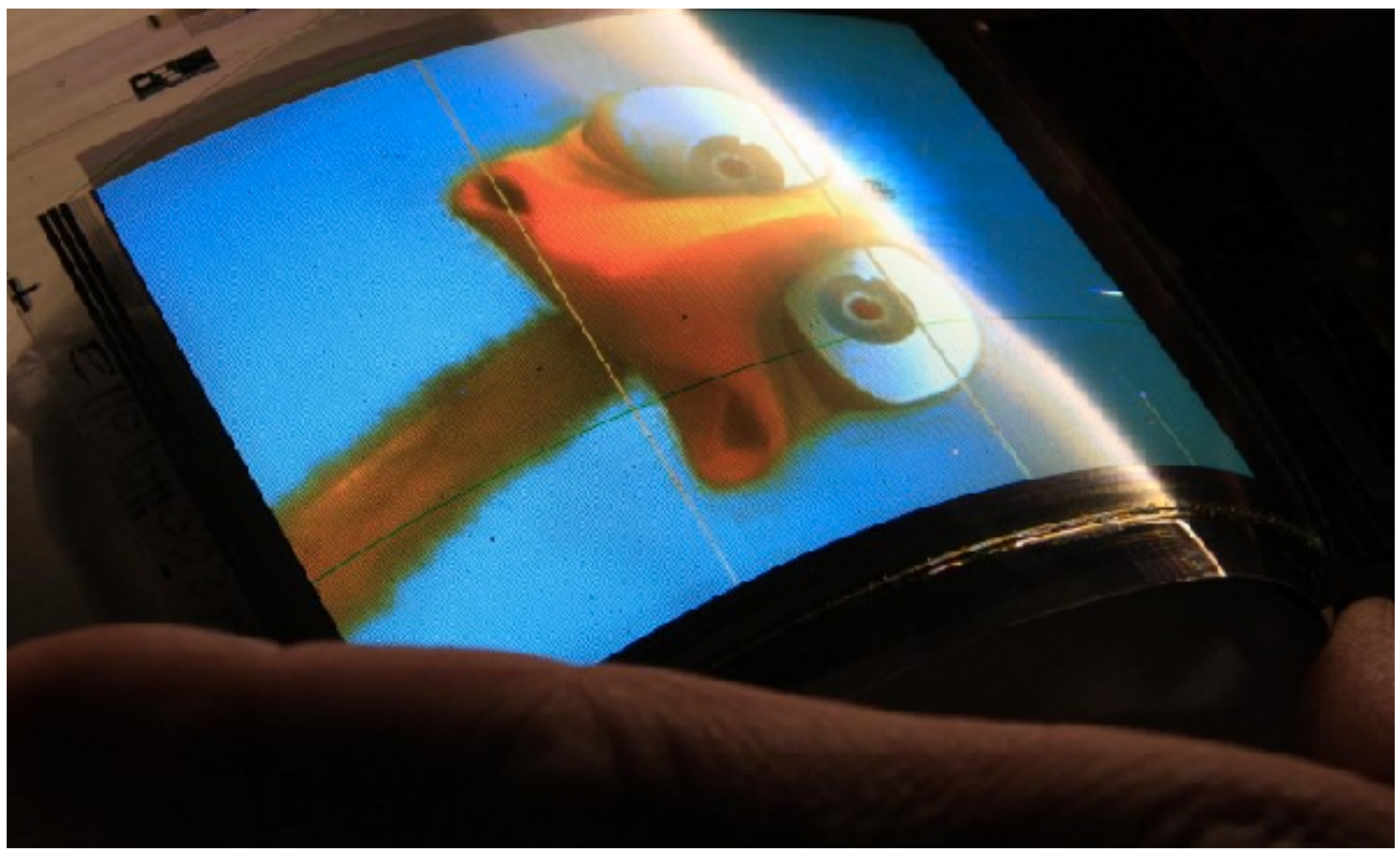


Image from: http://cdn.slashgear.com/wp-content/uploads/2013/04/flexible_oled-820x420.jpg

LCD vs OLED

- LCD displays: energy consumption is constant independent of the colours being displayed.
- OLED displays: energy consumption depends on the colours being displayed.

We can reduce energy consumption by choosing an appropriate GUI colour composition!

OLED Energy Consumption Optimisation Problem

- Consider an initial GUI design with its colours.
- Design variables: RGB colour of each pixel.
- Objectives:
  - Minimise power consumption on OLED displays.
  - Maximise contrasts between adjacent GUI components.
  - Minimise difference with respect to the original GUI design.
- Constraints:
  - Adjacent components of the GUI should not have the same colour or colours with too low contrast between them.

Optimisation Algorithm

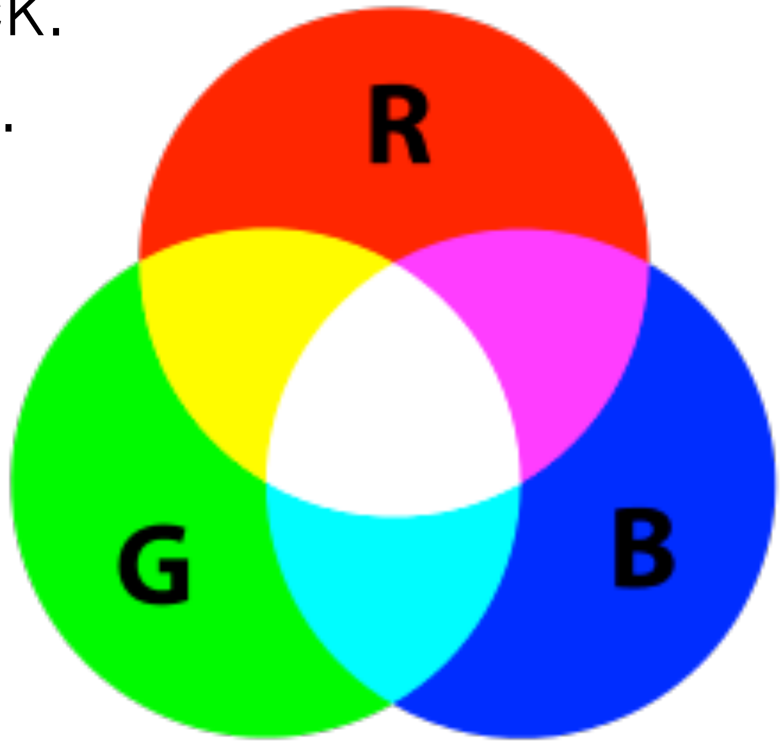
- NSGA-II has been used for this problem.
- The design for the algorithm will be explained at the same time as more details on the problem formulation are given.

Problem Formulation

- Consider an initial GUI design with its colours.
- **Design variables: RGB colour of each pixel.**
- Objectives:
  - Minimise power consumption on OLED displays.
  - Maximise contrasts between adjacent GUI components.
  - Minimise difference with respect to the original GUI design.
- Constraints:
  - Adjacent components of the GUI should not have the same colour or colours with too low contrast between them.

RGB Colour Scheme

- Produces colours by combining the primary (component) colours red, green and blue.
- Used to represent and display images in electronic systems.
- Each of the primary colours has a level of intensity.
  - Zero intensity for all colours gives black.
  - Full intensity for all colours gives white.
- Different numeric representations, e.g.:
  - (0-255, 0-255, 0-255)

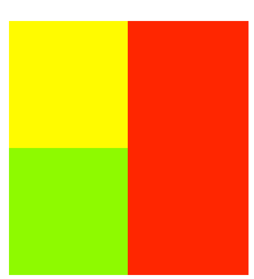


http://www.rapidtables.com/web/color/RGB_Color.htm

Image from: https://upload.wikimedia.org/wikipedia/commons/thumb/c/c2/AdditiveColor.svg/220px-AdditiveColor.svg.png

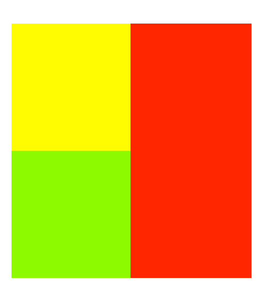
Bag-Of-Colour-Pixels (BOCP)

- When optimising power consumption, we can restrict the colours of the pixels.
    - Pixels with the same colour in the original GUI design should have the same colour in the optimised GUI design.
- Bag-of-colour-pixels (BOCP): collection of pixels with the same colour.
    - We have one BOCP for each different colour in the GUI.
    - Android View Server can be used to collect this information.



Bag-Of-Colour-Pixels (BOCP)

- When optimising power consumption, we can restrict the colours of the pixels.
  - Pixels with the same colour in the original GUI design should have the same colour in the optimised GUI design.
- Bag-of-colour-pixels (BOCP): collection of pixels with the same colour.
  - We have one BOCP for each different colour in the GUI.
  - Android View Server can be used to collect this information.



Bag-Of-Colour-Pixels (BOCP)

- When optimising power consumption, we can restrict the colours of the pixels.
  - Pixels with the same colour in the original GUI design should have the same colour in the optimised GUI design.

- Bag-of-colour-pixels (BOCP): collection of pixels with the same colour.
  - We have one BOCP for each different colour in the GUI.
  - Android View Server can be used to collect this information.

- We can exclude pixels from images to avoid changing the colours of these.

Representation

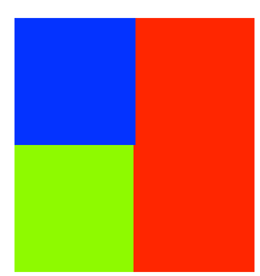
- Vector of genes, where each gene defines the colour of a BOCP.
- If we have N different colours in the original design, we will have N genes.

0-255,0-255,0-255	0-255,0-255,0-255	…	0-255,0-255,0-255

So, pixels with the same colour in the original design will have the same colour in the optimised design.

Representation

0,0,255	255,38,0	142,250,0



Initial Population

- Colour palette is used to create initial population to help creating appealing colour combinations.
  - The colour of each BOCP from the original design.
  - White.
  - Black.
  - Equidistant colour harmonies.
  - Equidistant monochromatic colours.

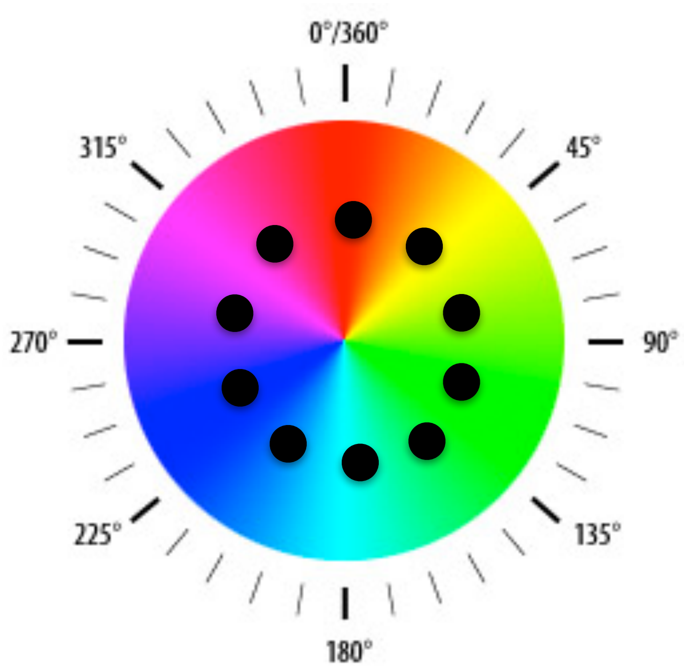


Image from: http://dba.med.sc.edu/price/irf/Adobe_tg/models/images/hsl_top.JPG

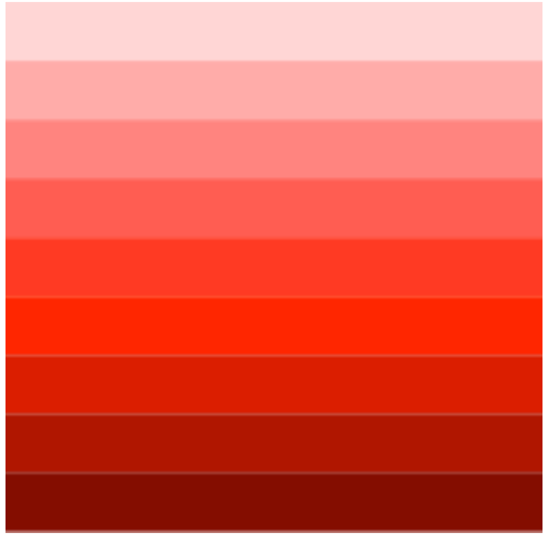


Image from: https://upload.wikimedia.org/wikipedia/commons/thumb/1/1a/CPT-Websites-monochrome.svg/201px-CPT-Websites-monochrome.svg.png

Initial Population

- Colour palette is used to create initial population to help creating appealing colour combinations.
  - The colour of each BOCP from the original design.
  - White.
  - Black.
  - Equidistant colour harmonies.
  - Equidistant monochromatic colours.

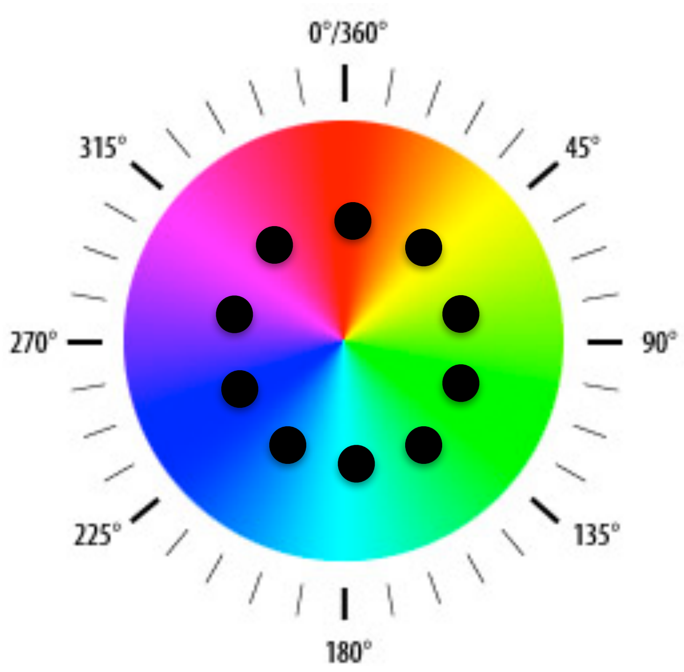


Image from: http://dba.med.sc.edu/price/irf/Adobe_tg/models/images/hsl_top.JPG

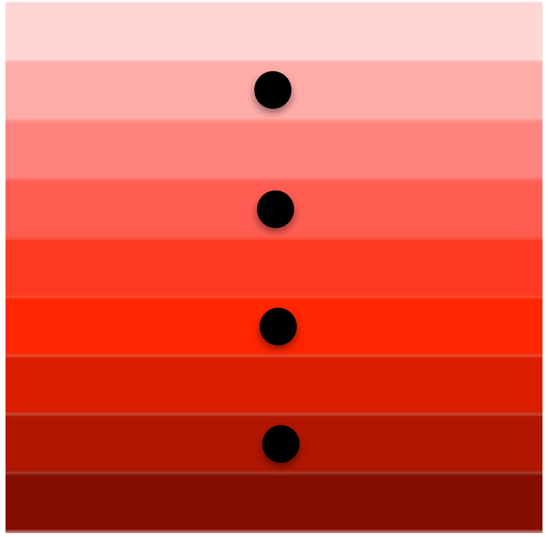
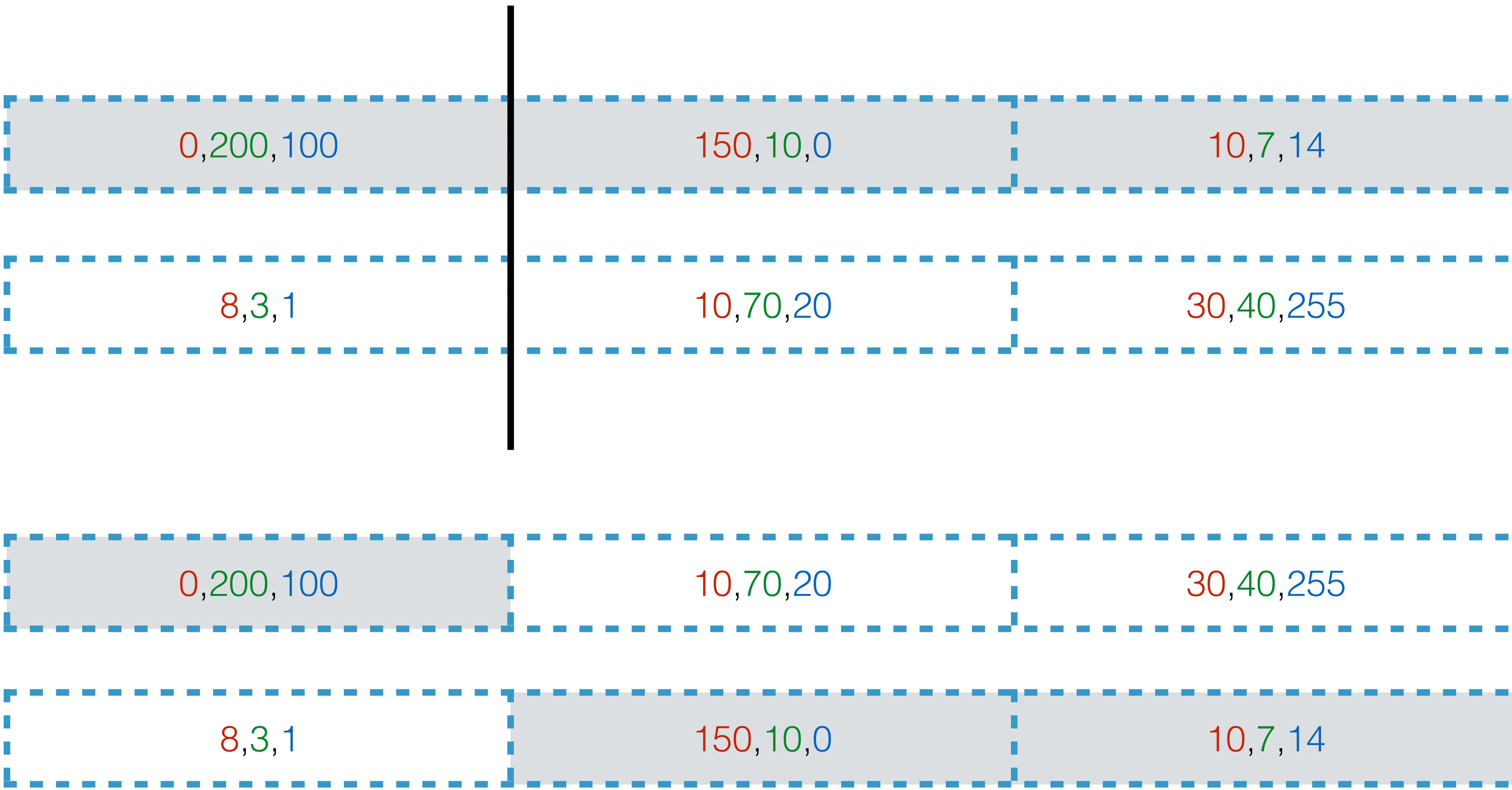


Image from: https://upload.wikimedia.org/wikipedia/commons/thumb/1/1a/CPT-Websites-monochrome.svg/201px-CPT-Websites-monochrome.svg.png

Crossover

- One-point crossover with probability *Pc*.



Mutation

- With probability *Pm*, change the colour of a gene to a new colour picked uniformly at random.

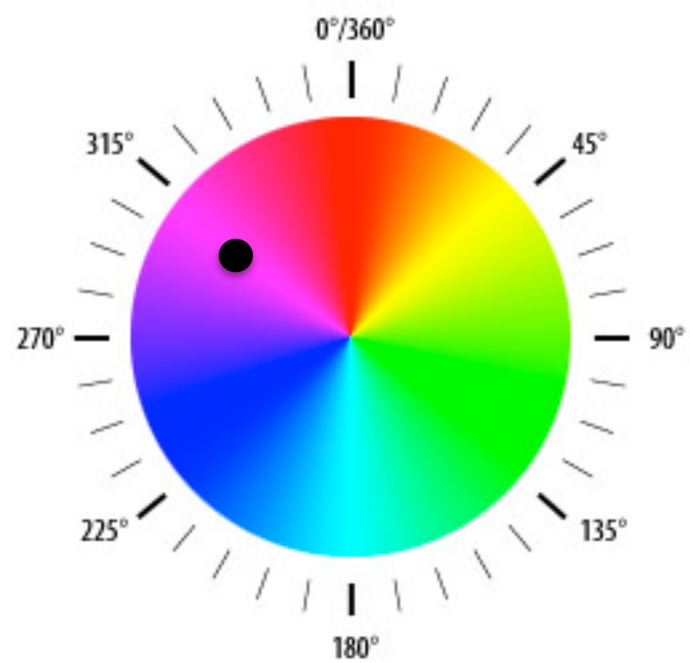


Image from: http://dba.med.sc.edu/price/irf/Adobe_tg/models/images/hsl_top.JPG

Problem Formulation

- Consider an initial GUI design with its colours.
- Design variables: RGB colour of each pixel.
- **Objectives:**
  - **Minimise power consumption on OLED displays.**
  - Maximise contrasts between adjacent GUI components. Minimise difference with respect to the original GUI design.
- Constraints:
  - Adjacent components of the GUI should not have the same colour or colours with too low contrast between them.

Calculating Power Consumption of a Pixel

- Power consumption of an OLED pixel is a function of the power consumption of its RGB component colours.
- Consider that $colour_{x,y,s}$ is the colour of pixel with coordinates x,y in a given screen s.
- The power consumption of this pixel is the following:

$$P(colour_{x,y,s}) = P_{<R>}(R_{x,y,s}) + P_{<G>}(G_{x,y,s}) + P_{<B>}(B_{x,y,s})$$

Power consumption of R component

Power consumption of G component

Power consumption of B component

Measuring Power Consumption

- The power of each colour component is screen-specific.
- One can measure the power consumption for each level of each component colour for a given type of screen.

https://www.msoon.com/LabEquipment/PowerMonitor/

Power Consumption Model

- We can’t measure the power consumption of each pixel during the evolutionary process!
- We need to get that information beforehand.
- Power consumption model for the screen used by Samsung Galaxy S4:

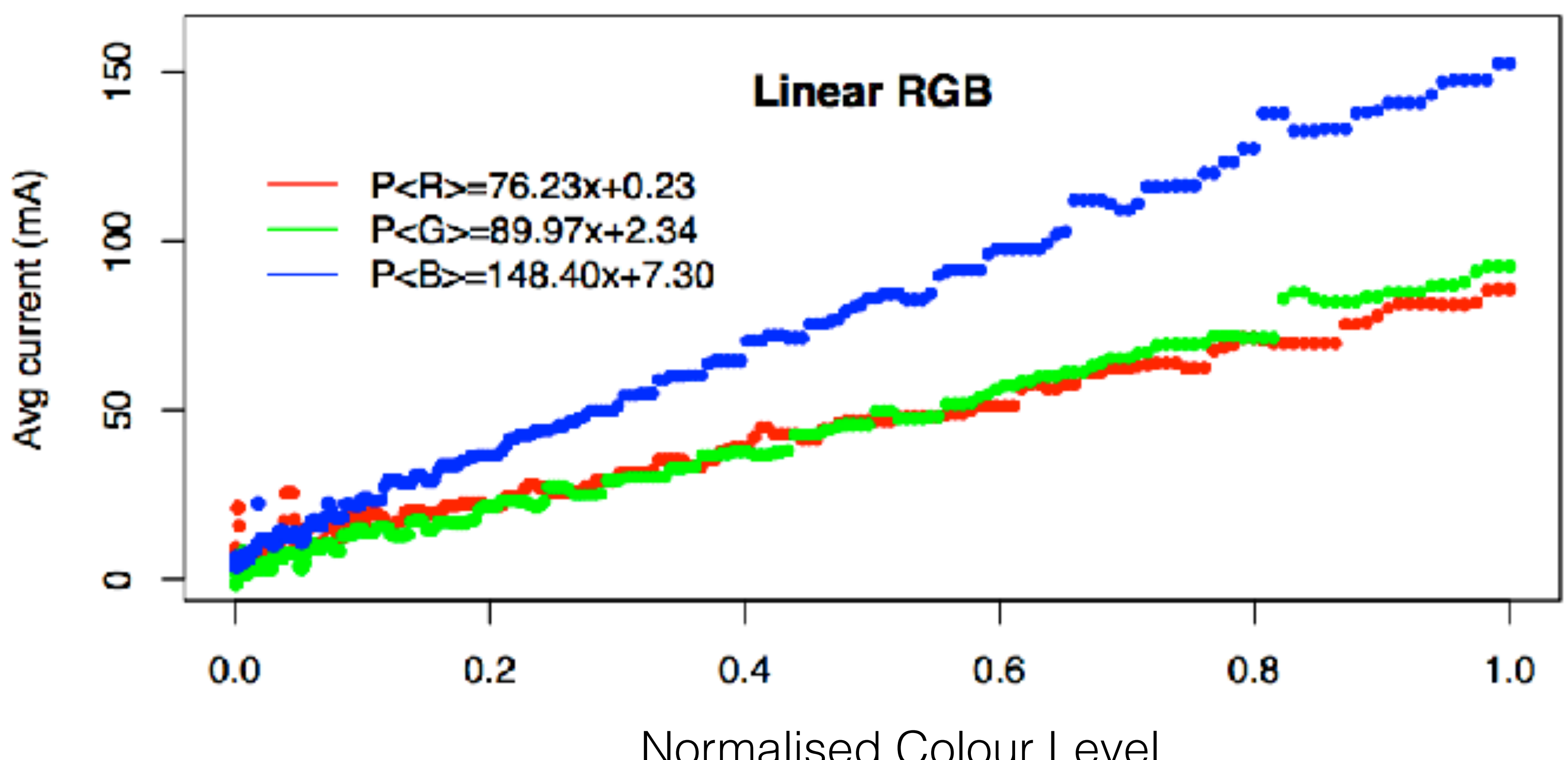


Image from Linares-Vasquez et al.

Calculating Power Consumption of a Screen

- A screen is composed of X * Y pixels, where X is the maximum x coordinate and Y is the maximum y coordinate of the screen.
- Power consumption of all pixels in a screen s:

$$\text{TotalPower}(s) = \sum_{x=1}^{X} \sum_{y=1}^{Y} P(\text{colour}_{x,y,s})$$

Calculating Power Consumption of Several Screens

- A GUI is composed of *S* screens.

$$\text{TotalPower(GUI)} = \sum_{s=1}^{S} \text{TotalPower(s)}$$

Calculating Power Consumption of Several Screens

- Screens that are used less time are less important.
- We can get information about the percentage of time a user spends on each screen by profiling application usage.

$$\text{TotalPower(GUI)} = \sum_{s=1}^{S} p_s \cdot \text{TotalPower}(s)$$

Percentage of time users spend on screen s

Problem Formulation

- Consider an initial GUI design with its colours.
- Design variables: RGB colour of each pixel.
- **Objectives:**
  - Minimise power consumption on OLED displays.
  - **Maximise contrasts between adjacent GUI components.**
  - Minimise difference with respect to the original GUI design.
- Constraints:
  - Adjacent components of the GUI should not have the same colour or colours with too low contrast between them.

Contrast Between Adjacent Components of a Screen

- Android View Server can provide us with the location of each component of a screen and its pixels.

$$\text{TotalContrast}(s) = \sum_{c=1}^{C_s} \sum_{c' \in adj(C_{c,s})} \textit{contrast}(C_{c,s}, C_{c',s})$$

where

- C_s is the number of components of screen s,
- $adjacent(C_{c,s})$ is the set of components adjacent to component$C_{c,s}$,
- contrast($C_{c,s}$, $C_{c',s}$) is the contrast between a component $C_{c,s}$ and component $C_{c',s}$.

Contrast For the Whole GUI

$$\text{TotalContrast(GUI)} = \sum_{s=1}^{S} \text{TotalContrast(s)}$$

where *S* is the number of screens.

Problem Formulation

- Consider an initial GUI design with its colours.
- Design variables: RGB colour of each pixel.
- **Objectives:**
  - Minimise power consumption on OLED displays.
  - Maximise contrasts between adjacent GUI components.
  - **Minimise difference with respect to the original GUI design.**
- Constraints:
  - Adjacent components of the GUI should not have the same colour or colours with two low contrast between them.

Calculating Distance to Original Design

- Distance between two colours: Euclidean distance between the component colours (RGB).

$$\text{distance}(RGB_1, RGB_2) = \sqrt{(R_1-R_2)^2+(G_1-G_2)^2+(B_1-B_2)^2}$$

Calculating Distance to Original Design

distanceDesigns <— 0

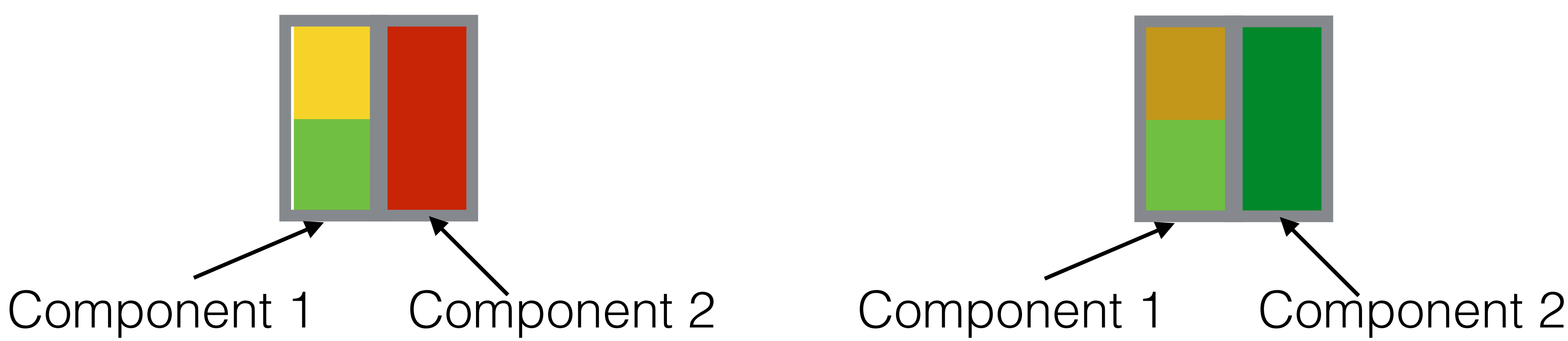
for each colour H in the new design

dist <— distance between H and its closest colour H’ in the original design.

If H and H’ involve components in common

dist <— 2 * dist // penalise the colour difference

distanceDesigns <— distanceDesigns + dist



Problem Formulation

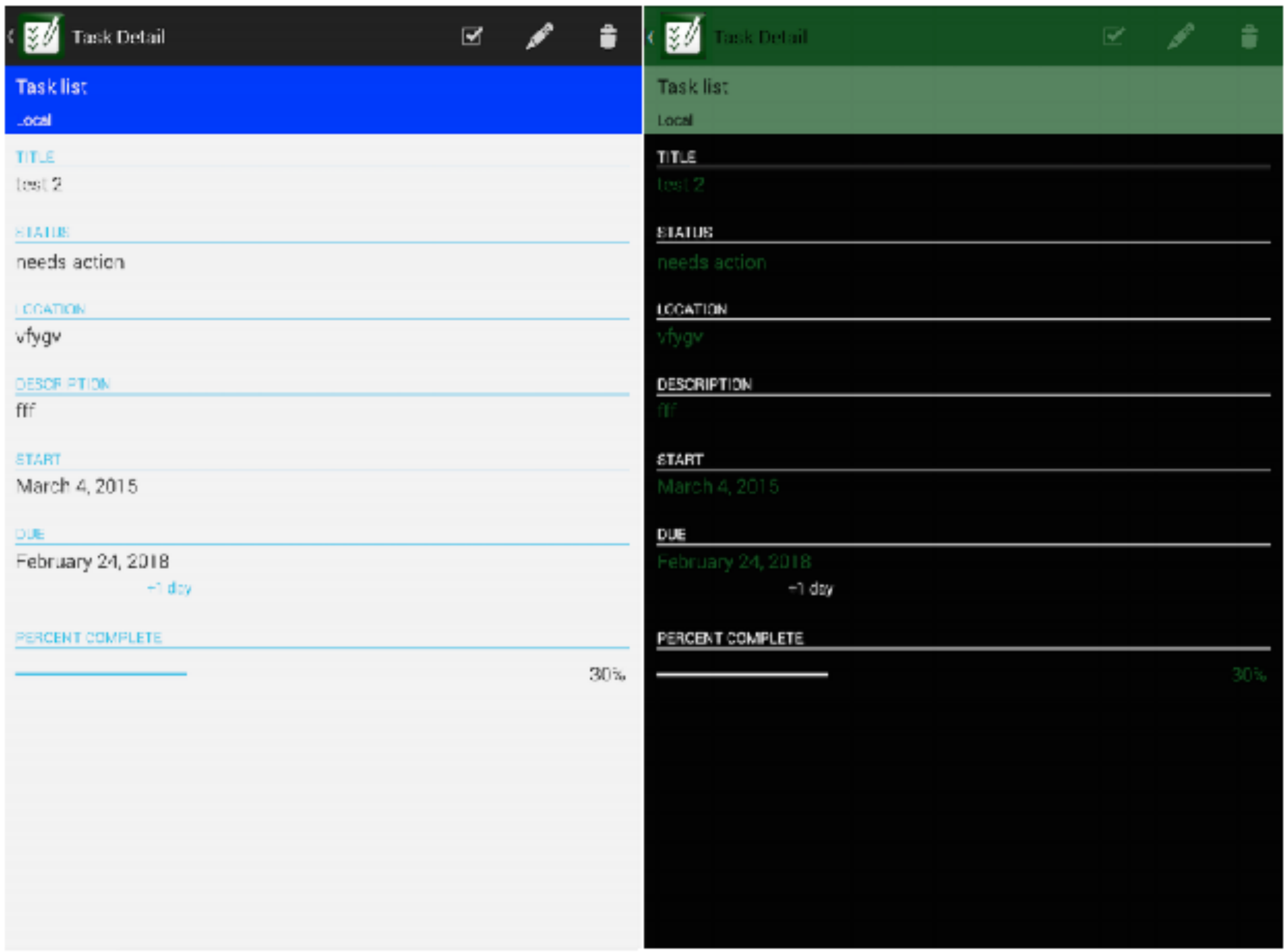
- Consider an initial GUI design with its colours.
- Design variables: RGB colour of each pixel.
- Objectives:
  - Minimise power consumption on OLED displays.
  - Maximise contrasts between adjacent GUI components.
  - Minimise difference with respect to the original GUI design.
- **Constraints:**
  - **Adjacent components of the GUI should not have the same colour or colours with too low contrast between them.**

Constraints

- Contrast between the colour of adjacent components of the GUI should be larger than a given threshold.
- Number of violations in a given solution = number of adjacent components with contrast lower than the threshold.
- Dealing with constraints: penalty based on the level of infeasibility.
  - Feasible solutions dominate infeasible ones.
  - Infeasible solutions with less violations dominate infeasible solutions with more violations.

Examples of Solutions

Images from Linares-Vasquez et al.



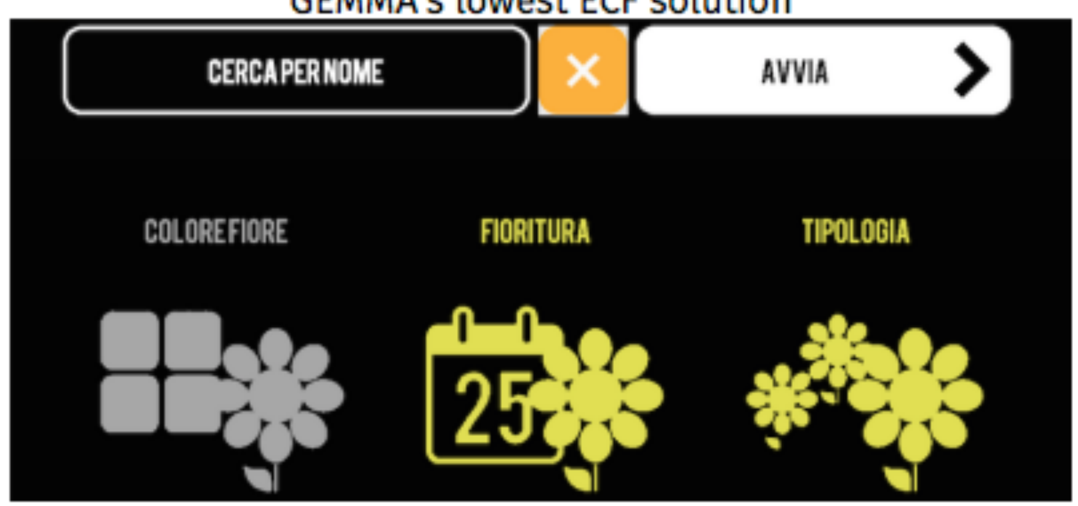
Original Design GEMMA's lowest ECF solution

The optimised GUI (right) offers energy consumption savings of up to 53% as well as an increase in terms of contrast ratio by 31%.

Examples of Solutions

Original Design

GEMMA's lowest ECF solution



Images from Linares-Vasquez et al.

Further Reading

Optimizing Energy Consumption of GUIs in Android Apps: A Multi-objective Approach

Mario Linares-Vásquez, Gabriele Bavota, Carlos Bernal-Cárdenas, Rocco Oliveto, Massimiliano Di Penta, Denys Poshyvanyk

Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering

Pages 143-154

http://dl.acm.org/citation.cfm?id=2786847&CFID=648846544&CFTOKEN=44813324