Amirkabir University
of Technology
(Tehran Polytechnic)

Assignment 1
Having Fun with Pixels!

Homeworks Guidelines and Policies

- ***What you must hand in.*** It is expected that the students submit an assignment report (HW1_[student_id].pdf) as well as required source codes (.m or .py) into an archive file (HW1_[student_id].zip).
- ***Pay attention to problem types.*** Some problems are required to be solved *by hand* (shown by the icon), and some need to be implemented (shown by the icon).
  Please don't use implementation tools when it is asked to solve the problem by hand, otherwise you'll be penalized and lose some points.
- ***Don't bother typing!*** You are free to solve by-hand problems on a paper and include picture of them in your report. Here, cleanness and readability are of high importance. Images should also have appropriate quality.
- ***Reports are critical.*** Your work will be evaluated mostly by the quality of your report. Don't forget to explain what you have done, and provide enough discussions when it's needed.
- ***Appearance matters!*** In each homework, 5 points (out of a possible 100) belongs to compactness, expressiveness and neatness of your report and codes.
- ***Python is also allowable.*** By default, we assume you implement your codes in MATLAB. If you're using Python, you have to use equivalent functions when it is asked to use specific MATLAB functions.
- ***Be neat and tidy!*** Your codes must be separated for each question, and for each part. For example, you have to create a separate .m file for part b. of question 3. Please name it like p3b.m.
- ***Use bonus points to improve your score.*** Problems with bonus points are marked by the icon. These problems usually include uncovered related topics or those that are only mentioned briefly in the class.
- ***Moodle access is essential.*** Make sure you have access to Moodle because that's where all assignments as well as course announcements are posted on. Homework submissions are also done through Moodle.

- ***Assignment Deadline.*** Please submit your work **before the end of April 10th**.
- ***Delay policy.*** During the semester, students are given 7 free late days which they can use them in their own ways. Afterwards there will be a 25% penalty for every late day, and no more than three late days will be accepted.
- ***Collaboration policy.*** We encourage students to work together, share their findings and utilize all the resources available. However you are not allowed to share codes/answers or use works from the past semesters. Violators will receive a zero for that particular problem.
- ***Any questions?*** If there is any question, please don't hesitate to contact me through the following email address: **ali.the.special@gmail.com**.

1. When Our Brain Can't Believe Our Eyes (12 Pts.)

Keywords: *Pixel Operations, Color Space, RGB Space*

Stare at the colored dots on Trump's nose in the image in Figure 1 for 20-30 seconds (the more, the better). Then immediately look at a blank wall and blink rapidly. Scary, huh?

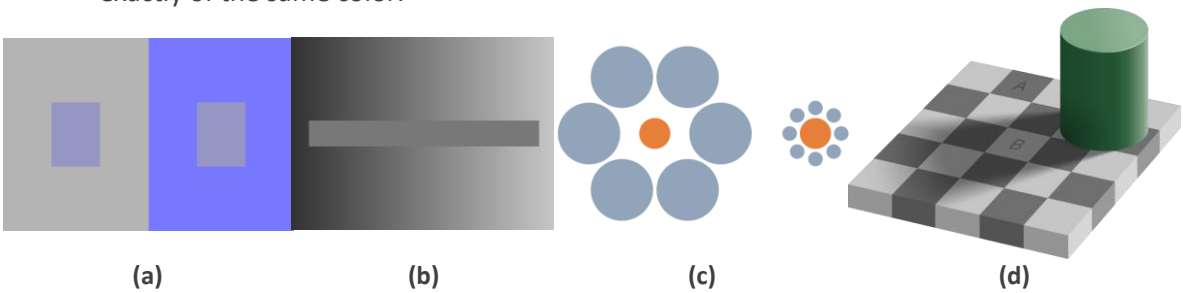
Figure 1 After staring at the colored points for around 30 seconds, this illusion will trick your brain into seeing angry Donald Trump on a wall! These types of optical illusions, known as "stare at red dots illusion", became a worldwide trend in September 2019 after a Facebook post (link).

Visual illusions similar to this occur due to properties of the visual areas of our brain as they receive and process information. In other words, our perception of an illusion has more to do with how our brain works, and less to do with the optics of our eye. An illusion is in fact a mismatch between the immediate visual impression and the actual properties of the object, and it often takes place when our brain's interpretation of the scene goes wrong.

In order to continue having fun, we decided to introduce more optical illusions to you (Figure 2) and see if you can use your image processing skills to prove they are not, in actuality, the same as they appear, and that our brains are not always reliable!

More precisely, each of the given images appear differently from the way they actually are. Your goal is to work on them and produce a new image in which the inaccuracy of our perception is proven.

a. **Color saturation illusion:** The two smaller squares in the larger gray and blue squares in the image appear to be blue or violet (left) and gray (right), while they are, in fact, the same color.
b. **Gradient optical illusion:** The horizontal bar in the middle appears to be brighter in the left and gradually becomes darker in the opposite side, while the truth is that the entire bar is of the same color.
c. **Ebbinghaus illusion:** Although the orange circle on the right appears larger than its peer on the left, they are exactly the same size.
d. **Checker shadow illusion:** It may seems impossible, but the squares marked A and B are exactly of the same color.



(a) **(b)** **(c)** **(d)**

Figure 2 Different optical illusions (a) Color saturation illusion (b) Gradient optical illusion (c) Ebbinghaus illusion (d) Checker shadow illusion

Hint: In some cases, the ambiguity will be resolved by just unifying background color. Sometimes you need to remove the objects or colors that you think may be the main reason behind the illusion.

Note: You are only allowed to complete these tasks using simple pixel operations.

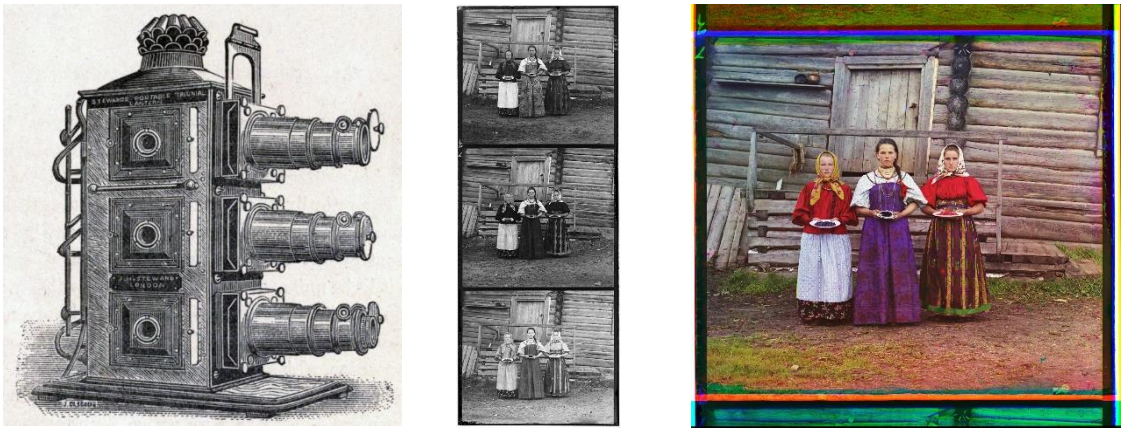
Recommended MATLAB functions: `find()`, `minus()`

2. Bringing A Century-Old Dream Into Reality (12 Pts.)

Keywords: *Color Photography, RGB Channels, Image Alignment, Resolution Pyramid*

Immediately after the first ever camera photograph **"***View from the Window at Le Gras"* (here) was recorded, researchers started to dream about **Color Photography**, i.e. adding real-world colors to the photos. One of the early pioneers in this area was Sergey Prokudin-Gorsky (1863-1944), a Russian chemist and photographer who travelled the Russian Empire from 1909 to 1915 and documented thousands of invaluable photos of that era.

Although color photography wasn't introduced at that time, Prokudin-Gorsky devised a method to capture colors for the photographers of the future. To do so, he captured three different grayscale pictures of every scene, each with a red, green and blue color filter in front of the camera. He then printed the resultant monotone photo-in-triplicate (Figure 3, part b.) as a positive and projected it through a three-lens lantern, each using the same colored filters (Figure 3, part a.). Although Prokudin-Gorsky managed to effectively present full color images on screen, he never saw his printed colored photos. Yet his legacy led to producing hundreds of high-quality color images of century-ago Russia which is now housed by the Library of Congress (here).



(a) **(b)** **(c)**

Figure 3 A brief depiction of Prokudin-Gorsky's method to generate color photos (a) Three channel lantern projector similar to the one used by Prokudin-Gorsky (b) An example of vertical images taken by him in three channels, from Alim Khan, The Emir of Bukhara in 1911 (c) The result obtained by merging the channels together

Here, you are given some of Prokudin-Gorsky's grayscale image composites, Figure 4. These triple-framed images contain three grayscale photos taken in the early 1900s, and each frame represent the image captures with a blue, green and red filter. Choose four composites to your liking and use them as inputs in the following parts.

a. **Preparation.** Write a function `extract_channels()` which takes a triple-framed image as input, chops it vertically into three separate parts and save these channels with appropriate names.
b. **Merge.** Implement another function, `stack_channels()`, which takes three channels as input and stack each of the three images into a proper color channel and display a single colored image. It is expected that the stacked photos look wonky and unaligned. Include it in your report.
c. **Alignment.** Now write a third function, `align_results()`, which takes the image you obtained in the previous part and align it appropriately. This function must search over possible pixel offsets in the range of [-30, 30] to find the best alignment for each channel. One simple way to do so is to keep one channel fixed, and align the other two by searching

over the offset range both horizontally and vertically. Pick the alignment that maximises a similarity metric (of your choice) between the channels, e.g. dot products, normalized cross correlation, etc.

Note: For full credit, your report needs to include properly aligned images. Find a similarity metric that will accomplish this.

d. **Resolution Pyramid.** For large offsets and high resolution images, it can be computationally intensive to compare all the alignments for a specific range of displacements (e.g. [-30, 30]). It is often advised to start by estimating an alignment on a low-resolution version of the image before applying it on higher resolutions. In this part, you are asked to implement a two-level image pyramid by scaling the extracted channels down by a factor of 2, and then execute your alignment over the range of offsets [-15, 15]. After choosing the best alignment based on your similarity measure, use it as a starting place to again run the alignment in a small range [-15, 15] in the full resolution images.

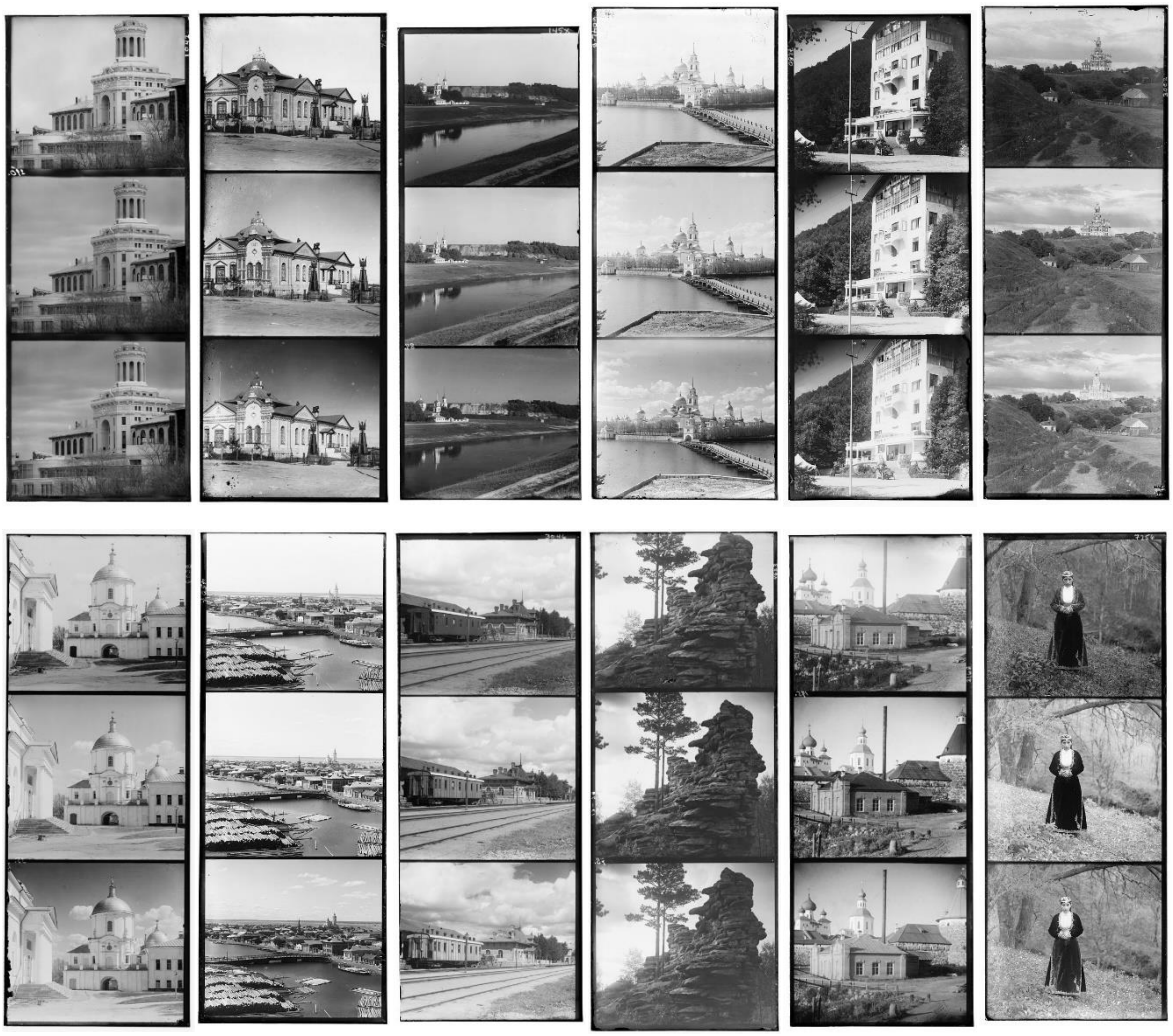


Figure 4 A dozen of Prokudin-Gorsky's negatives given here as the input. Although many of his images were lost, the majority of them are now kept in the U.S. Library of Congress

3. Blue or Gold? Black or White? Let's End This Debate Once and for All! (12 Pts.)

Keywords: *Color Quantization, Image Segmentation, Clustering, K-Means Algorithm*

One week before the wedding of Grace and Keir Johnston, a couple from Scotland, the bride's mother took a photograph of a dress which she planned to wear at the wedding (Figure 5) and sent it to her daughter. After some dispute over color of the dress, the bride decided to post the image on Facebook, where her friends also disagreed over the color; some saw it as white with gold lace, while others saw it as blue with black lace. Few days after the wedding, on 26 February 2015, a friend of the bride and groom posted the image to her blog on Tumblr, which led to further discussion about the true perceived color of the dress.

Figure 5 The image may seem to be a normal photo taken from a simple dress, but became globally famous for its ambiguous colors. Some people believe it consists of blue and black, while others believe its main colors are white and gold.

The image quickly went viral through the internet and became the subject of millions of debates. It not only broke the internet, but also drew the attention of many image scientists and researchers. Although the color of the dress was eventually confirmed as black and blue, yet there is no consensus explanation of the reasons why the dress appears differently among the viewers.

This image, known as *The Dress*, is given to you along with three other images of this type (Figure 6), each with debatable colors. Your goal is to use image processing techniques in order to find the most frequent colors in each image and determine which side of these arguments are correct. Perform the following algorithm, known as K-Means, on each of the given images and clear up any ambiguity about their colors!

- Determine the number of colors needed to be extracted from the image, K.
- Randomly select K pixels as the centroids (not necessarily from the image).
- Assign each pixel to their closest centroid and form K clusters.
- Compute the new centroids by averaging the pixels in each cluster.
- Reassign each pixel to the new closest centroid. Repeat the previous part until no reassignment takes place.

a. Set $K = 2, 3, 5, 7$, and report the most frequent colors in each images.
b. Perform **Image Segmentation** by assigning each pixel to its closest color among those extracted in the previous part. Include the images in your report (16 images in total).

(a)

(b)

(c)

Figure 6 People perceive colors in these images differently (a) Another image which, similar to The Dress, became a social media trend. Some see gray and teal as the main colors of the outfit, whereas the others see pink and white (b) Some may perceive the strawberries as being red, while there's not a single red pixel in the image (c) The boxes appear to be bluish, while they're not

4. Where's Wally? (12 Pts.)

Keywords: *Image Thresholding, Template Matching*

"Where's Wally?" is a British puzzle book series which consist of a set of elaborate illustrations depicting dozens or more people doing a variety of funny things at a certain location. The challenge is to find a specific character called Wally (or Waldo in the U.S.) hidden somewhere in the image. Wally is recognised by his red-and-white-striped sweater, blue jeans, bobble hat and round glasses.

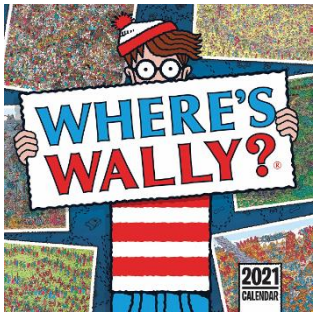


Figure 7 Wally's appearance is unchanged among all the puzzles, which is in fact the main clue to identify him between the crowds

The aim of this problem is to get you familiar with the subject of **Image Thresholding** by solving a couple of "Where's Wally?" puzzles. Bearing in mind the fact that Wally always wears a shirt with the same pattern, one can easily restrict the search area to those regions who are more similar to his shirt. In other words, the idea is to keep only red and white pixels in the images and perform the search in those areas.

You are provided with four different "Where's Wally?" puzzles. In each puzzle, your goal is to find the whereabouts of Wally and display his location in the original image. You are also given a few sample images of Wally (Figure 8) which you may find useful.

Figure 8 Although his costume is always the same and might look similar to other characters, Wally may be carrying a stack of books that vary from scene to scene (a) Wally (b) His lookalikes

a. Find two approximate RGB values for the red and white parts of Wally's shirt.
b. Try to preserve those pixels of the puzzles similar to Wally's shirt by setting appropriate thresholds using the values you found in the previous part. You may also need to keep a specific range of pixels around them, e.g. a 50×50 pixels square. Set the values of other pixels to zero (black), and display the results you obtained for each puzzle.
c. Search for Wally in the modified puzzles. You only need to search in non-black pixels, i.e. those who were detected similar to Wally's shirt.
d. Display each puzzle with Wally's position indicated by a bounding box (rectangular borders around his coordinates).

Note: There are also several Wally lookalikes in each puzzle. Try not to be fooled by them.

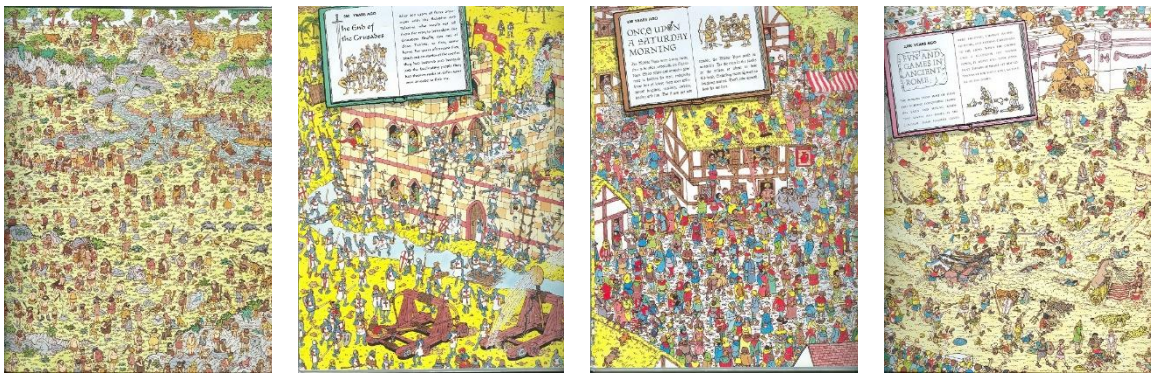


Figure 9 Different puzzles of "Where's Wally" with Wally hiding somewhere in the scene

5. Demographists and Statisticians Would Love Image Processing! (18 Pts.)

Keywords: *Basic Operations on Images, Color Space, RGB Space, Pixel Operations*

If you think the materials you've learned so far are insufficient for implementing real-world applications, well, think again. Because in this problem we're going to see how these simple techniques can come into use to interpret complicated demographic images and plots.

First, consider the map in Figure 10 which displays internet censorship and surveillance status across the world in 2018. Considering the legend in the table, answer the following questions:

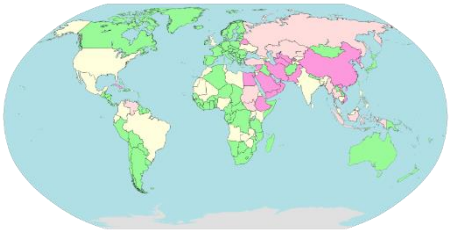


Figure 10 Internet censorship and surveillance by country (as of 2018)

Color	RGB values	Meaning
	(255,153,221)	Pervasive censorship
	(255,221,221)	Substantial censorship
	(255,255,221)	Selective censorship
	(152,251,152)	Little or no censorship
	(224,224,224)	No data

a. According to this map, what is the percentage of water on earth?

   Hint: The RGB values for the outer regions are `(255,255,255)`.

b. What is the percentage of countries with pervasive censorship?

c. What is the percentage of countries which apply selective or little or no censorship?

Next, consider another map given in Figure 11, which depicts provinces of Iran by population in 2014. The legend of the map is also included in the figure.

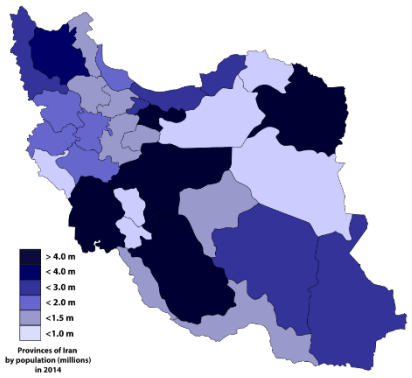


Figure 11 Provinces of Iran by population in the year 2014

d. Calculate the percentage of provinces with more than 4 million population.

e. Calculate the percentage of provinces with more than 2 million population.

f. What is percentage of provinces with less than 1 million population in the eastern part of the country?

g. Approximately compare the northern provinces with southern provinces in terms of population.

Now let's deal with a more complicated case. The evolution of Iran population divided into urban and rural areas is shown in Figure 12.

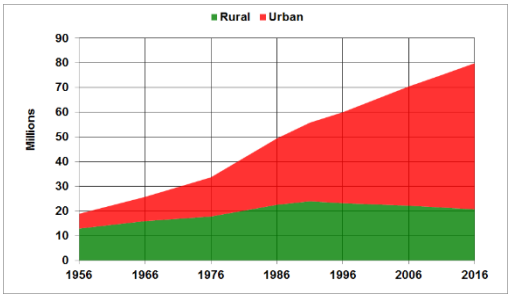


Figure 12 Population of Iran between the year 1956 to 2016, divided into urban and rural populations

h. Find the ratio between the number of urban residents to the total population of the country over the given period.

i. What was the percentage of rural residents between the years 1996 to 2006?

j. Calculate the exact number of city dwellers in the decade started from the year 1976.

k. Since when the population of urban residents started to be more than the population of rural areas?

Finally, you are given Iran population pyramid in the year 2020.

l. Do women outnumber men in Iran? Calculate each gender's population.
m. What is the percentage of those over 50?
n. Compare the population of male and female under 20.
o. Find the exact number of men in their 40's (between 40 and 50 years old).

Note: The surplus populations indicated with darker colors must be ignored in calculations.

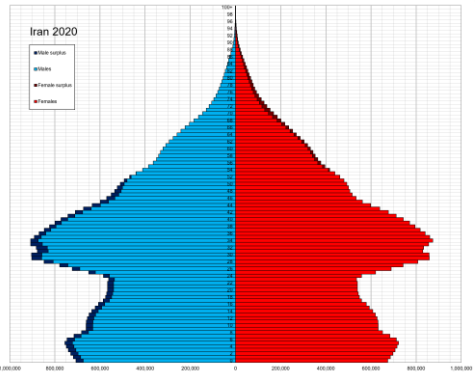


Figure 13 Iran population pyramid in 2020

Note 1: You are required to use low-level pixel operations to find the whereabouts of specific regions in the images. Using functions like `imtool()` (in MATLAB) is not allowed.

Note 2: Since the colors are blurry in boundary regions, you need to either apply thresholding before the calculations or consider a range when counting pixels.

Note 3: In some cases the calculations might not be accurate due to the presence of a legend in the images.

Recommended MATLAB function: `find()`

6. Let's Play Tetris! (24+10 Pts.)

Keywords: *Basic Operations on Images, Color Space, Pixel Operations, Image Rotation, Frame, Video Sequence*

Tetris is arguably one of the most popular video games ever made, with over 495 million copies sold in all consoles, which makes it the second best-selling video game franchise after Super Mario. Its mobile version, created by Electronic Arts, has also been sold over 100 million copies, making it the third best-selling game ever made, only after Minecraft and Grand Theft Auto V.

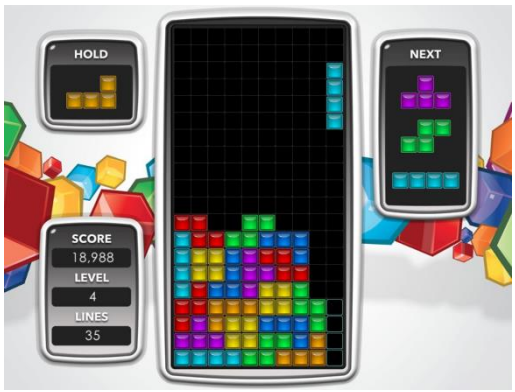


Figure 14 One of the many variations of the game. Tetris is available on over 65 platforms, setting a Guinness world record for most ported video game title ever made.

The game is built on simple rules: players have to complete lines by moving different shaped pieces, known as *tetrominoes*, which descend onto the lower parts of the screen. During this descent, the player can move pieces laterally and rotate them until they touch the bottom. The goal is to use these pieces to create as many horizontal lines of blocks as possible. Once a line is completed, it disappears and the players earn some points and proceed to fill the remaining vacated spaces. When the playing field is filled, the game ends. You can use this link to play an online version of the game.

In this question, you are provided with a 1200x2000 image of a classic version of the game in a certain state, along with five different tetrominoes (Figure 15). The image contains blocks of the size 100x100, hence there are 12 blocks in horizontal and 22 blocks in vertical directions (of course the playable field doesn't include the boundary blocks depicted with gray). We define a mini game in which these five pieces must be played respectively. The objective is to display each of these five moves separately, from their arrival to the game to their final states (at most 20 frames), according to the following rules:

	Rules of the game
1	Pieces enter the screen from the middle
2	Each piece has specific initial state (illustrated in Figure 15)
3	Pieces move block by block until they touch other pieces or the bottom surface
4	In addition to vertical moves, pieces can move one block to the left or right
5	In addition to vertical moves, pieces can be rotated clockwise
6	If all blocks in one or more lines are filled, the line(s) will be removed and the upper blocks will be shifted down
7	In the end, at least 5 lines must be completed and removed

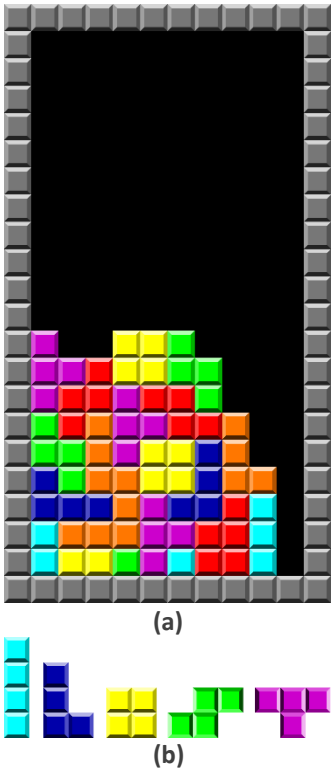


Figure 15 Images given as input of the problem (a) A Tetris game in a middle state (b) Different pieces known as 'I', 'L', 'O', 'S' and 'T'

a. Play with 'T' and display all the intermediate states as well as the final one.
b. Play with 'L' and display all the intermediate states as well as the final one.
c. Play with 'S' and display all the intermediate states as well as the final one.
d. Play with 'I' and display all the intermediate states as well as the final one.
e. Play with 'O' and display all the intermediate states as well as the final one.
f. Use the frames obtained above and create a video sequence of all the moves. Consider a 0.5 second pause between each move.

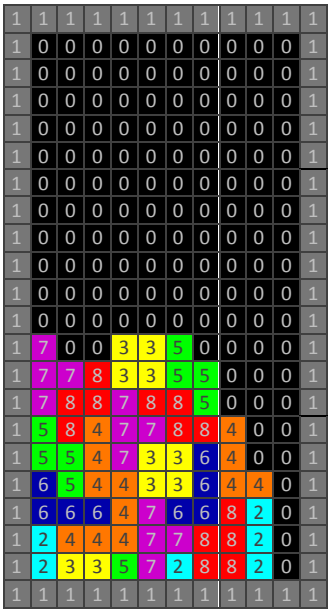


Figure 16 One approach is to represent each state of the game with a matrix in which each color is denoted by a specific number

Hint: One may take different approaches in order to display the movements of each piece in the game. One way is to consider each state of the game as a matrix (like in Figure 16) and implement a function which takes matrix `S` as a state, string `p` as a piece type, integer `t` as a piece state and tuple `(px,py)` as block coordinates, and displays the current state of the game. The other is to implement a function which takes image `S` as a state, image `p` as a piece, and tuple `(px,py)` as block coordinates, and displays the current state of the game. In each case a function must also be implemented to perform piece rotations.

Note: You don't need to implement the rules in your code. In other words, you are only asked to display each state of the game, and the purpose of this problem <u>is not</u> the implementation of a real Tetris game.

Amirkabir University of Technology
(Tehran Polytechnic)

7. Some Explanatory Questions (5 Pts.)

Please answer the following questions as clear as possible:

a. Explain how the optical illusion in Figure 1 works.
b. Two sets of images are available. Set 1 contains two images, image A is of the size 400x300 whereas image B is of the size 1600x1200. Set 2 also contains two images, image C with 8-bit and image D with 24-bit color depth. Compare the images in each set with each other in terms of quality.
c. Is grayscale to RGB conversion possible? Justify your answer.
d. Why do we have different color spaces? Do they have specific advantages/disadvantages over each other?
e. Explain the procedure of digitizing a continuous image in detail. Provide example if necessary.

Good Luck!
Ali Abbasi