Игорь:
Good morning to all the future IT specialists and all who have already stepped on this tough road. My name is Igor Shalygin. This is Anton Tsoy and Kirill Kozlov. We are going to introduce you to our project Indoor Slam.

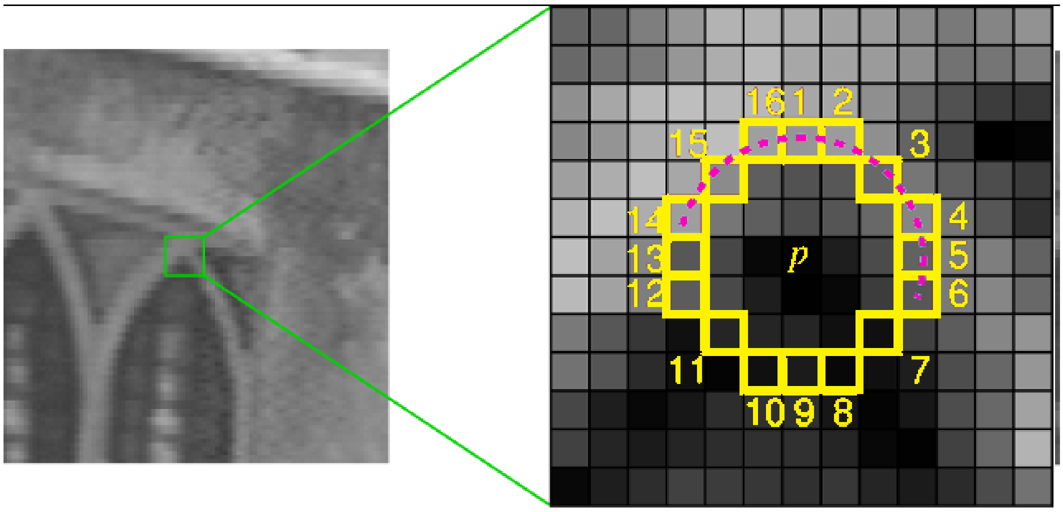
Антон:
Well, Indoor SLAM is a method of constructing the most accurate indoor map by processing video captured by a moving camera. In simple terms, we want to build a 3D map of the room based on video from a monocamera.

Кирилл:
The goal is to implement a main approach of Indoor-SLAM algorithm. We have chosen the OpenCV library in C++ to get the most effective implementation.

Our tasks:

- FAST extractor implementation

    Sooo, let's talk about the tasks we completed this semester. The first thing we needed to do was the feature extractor.
    Feature is a distinct and recognizable pixel in a frame that the algorithm can identify and track as a key point in the following frames. These key points that we extract will become the skeleton of our 3D map in the future.
    To extract features, we have chosen a FAST algorithm. It compares the intensity of a feature candidate P with each point from a 16-point circle that surrounds P, as shown on the slide. Then, a feature is detected at P if the color intensities of next pixels are all above or all below the intensity of P by some threshold.



- Implement feature tracker

  When we've extracted all the features we need to track them.
  To track a feature means to find two corresponding coordinates of this feature in the first and second frames.
  We can do it pretty easily using the Least Sum Difference algorithm.
  <u>next slide.</u>
  At first,We propose the feature in the next frame moved not much then sigma(on the slide). This magic value was found judging by a lot of tests.
  <u>next slide.</u>
  We find the search region by using these simple formulas.(on the screen)
  Look scary, but, in fact, they are pretty obvious.
  <u>next slide.</u>
  Then for each pixel within this region in the second frame we compute the Sum of squared differences between the search region and the circular batch around this feature and then the batch with the least sum will contain our feature in the middle.
  <u>next slide.</u>
  Obvious formulas.
  <u>Next slide.</u>

Here you can see an example of ft on a simple object. First picture contains extracted features and the Second picture contains tracked.

- <Кирилл>
    - On the next step we need to get rotation and transition between extracted and tracked points vectors which will be used for placing 3D points in the world system.
    - Here you can see some examples
    - Firstly, for this step we need to have a calibration matrix which contains camera intrinsic parameters essential for correct computations. This matrix can be found using photos with object with known 3D points positions.
    - Combining this matrix with corresponding extracted and tracked points we can estimate the essential matrix.
    - Using Singular values decomposition, essential matrix can be decomposed into 4 combinations of rotation and transition and one of them represents moving of points from the first frame to the second
    - Best combination will project maximum points at the front. To find this one we use linear triangulation and check the count of 3D points at the front of the camera.
- Triangulation

    The last step at this moment is a triangulation. So, triangulation is the process of determining a point in three-dimensional space given its projections onto two or more images.
    For triangulation we use things estimated earlier. We've gotten two-dimensional coordinates of key points on the first frame and corresponding tracked points on the second frame and projection matrix between them.

<u>Написать связку со следующей частью</u>
В общем, я накидаю сюда те штуки, которые, как мне кажется, визуально могли бы быть интересны:

- *Конструируем из точек и матриц отображения (с черточкой - относятся ко второй матрице / второму массиву точек, верхние индексы - ряды) вот такую матрицу*

$$\mathtt{A} = \begin{bmatrix} x\mathbf{p}^{3\top} - \mathbf{p}^{1\top} \\ y\mathbf{p}^{3\top} - \mathbf{p}^{2\top} \\ x'\mathbf{p}'^{3\top} - \mathbf{p}'^{1\top} \\ y'\mathbf{p}'^{3\top} - \mathbf{p}'^{2\top} \end{bmatrix}$$

- *Решаем SVD вот такого вида:* $A = USV$
- *Последний столбец (четвёртый) в V – это и будет наша трёхмерная точка в гомогенных координатах*
- *На слайде это можно было бы показать стрелками от построения матрицы A к записи SVD, а потом обозначит (может также стрелочкой), где находится наша новая точка*

- Visualization
  Okay, let's talk about the visualization part. We use a special tool called PyBullet to visualize three-dimensional points we have computed.

<u>Игорь:</u>
As a result, we want to make a program which can construct an indistinguishable from reality 3D map of a room, or just dense at least.

<u>Кирилл</u>
Задачи на следующий семестр:

- Now, we would like to outline our goals and tasks for next semester. So far, we are using basic triangulation from the OpenCV library. However, we have already realized that we are not satisfied with this solution. We will work on a more accurate and suitable algorithm for us.
- We are also thinking about optimizing the performance of our software. The first thing we can do is to add multithreading <какую?>. (Наладить взаимодействие визуализатора и триангуляции не через файл, а через pipe)
- The fourth important task is to implement detection of cycle paths. This is necessary for correcting computed 3D points which got earlier.

<u>Игорь</u>
Here you can see an amazing CV book and great article we used to make a deep dive into the SLAM problem. Thanks for your attention.

Возможные вопросы:

1. Что такое essential matrix?
2. Что такое SVD?
3. Чем ваш проект отличается от проекта Outdoor?
4. Что вы использовали готового из OpenCV и прочих библиотек, и что сделали сами?