

Ремонт механизма

Евгений Шальнов, Антон Конушин

Срок сдачи: 2 октября (23:59).

1 Обзор задания

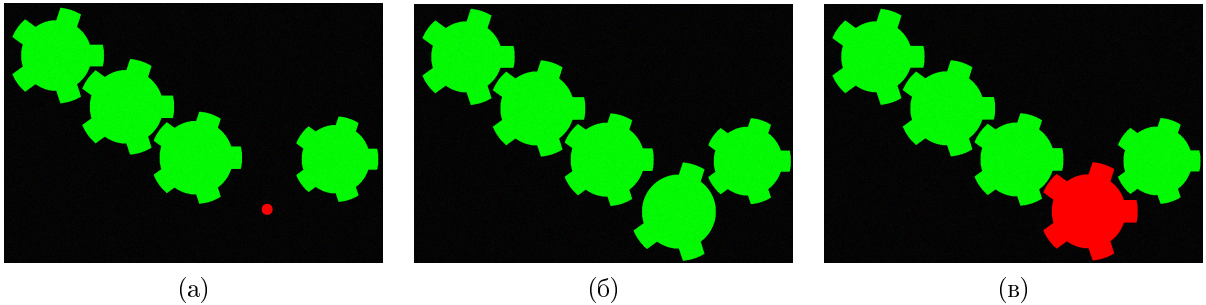


Рис. 1: (а) входное изображение базовой части; (б) входное изображение бонусной части; (в) результат работы алгоритма

Еще до того, как компьютеры стали электронными, люди очень хотели быстро считать. Поэтому были созданы механические счетные машины такие, как эта. К сожалению, их шестеренки часто ломались. Ваша задача — разработать программу, которая будет находить и заменять сломанные шестеренки на изображении механизма.

На изображениях находятся 3 типа объектов:

- исправные шестеренки;
- сломанная шестеренка;
- ось для шестеренки.

2 Базовая часть задания

За базовую часть задания можно получить **5** баллов, однако эта часть должна быть выполнена полностью.

Эта часть задания состоит в установке недостающей шестеренки на имеющуюся ось (см рис. 1(а)). Вам необходимо выбрать одну из трех предложенных Вам шестеренок расположенных в файлах с названием `<имя_входного_файла>_<номер_варианта>.bmp`

Обратите внимание, что в базовой части нет сломанных шестеренок. Этапы решения задачи:

1. Бинаризация изображения (например, с помощью отсечения значений пикселей по порогу).
2. Выделение связанных компонент на изображении (с помощью последовательного сканирования или обхода в ширину).
3. Классификация связанных компонент на шестеренки и ось.
4. Вычисление геометрических признаков связанных компонент. К таким признакам относятся:
 - положение центра шестеренки;
 - радиус меньшей окружности, образующей шестеренку;
 - радиус большей окружности, образующей шестеренку.
5. Выбор шестеренки, которую необходимо установить на ось, учитывая ее размер (все шестеренки уже повернуты необходимым образом).

Можно считать, что радиус большей окружности, образующей шестеренку, — расстояние от центра до самой удаленной точки шестеренки, а радиус меньшей окружности — расстояние от центра шестеренки до ближайшей точки, не принадлежащей шестеренке.

3 Бонусная часть задания

За бонусную часть задания можно получить до **10** баллов, при условии выполненной базовой части. Она состоит из двух частей, каждая из которых оценивается максимум в **5** баллов.

3.1 Поиск сломанной шестеренки (до 5 баллов)

В отличие от базовой части здесь нет оси для установки шестеренки, но одна из шестеренок сломана (см рис. 1(б)). Предлагается найти и заменить эту сломанную шестеренку.

Для выполнения этого задания Вам может потребоваться метод под названием `distance transform`. Подробное описание эффективной реализации можно найти здесь (см. раздел 2). Реализация этого метода занимает не более 20 строк кода на C++. Использовать готовые реализации нельзя (да, мы проверяем на плагиат)!

Не забудьте заменить сломанную шестеренку. За каждую верно замененную шестеренку начисляется по 1 баллу.

3.2 Определение количества зубцов шестеренки (до 5 баллов)

Здесь Вам предлагается также определить количество зубцов у каждой шестеренки на изображении (в том числе, у сломанных, если они присутствуют).

За каждое изображение, где верно найдено количество зубцов всех шестеренок, добавляется **0.5** балла.

4 Использование сторонних библиотек

В данном задании Вам необходимо самим реализовать всю обработку изображений, то есть заимствовать ничего нельзя. Вы можете использовать только стандартные библиотеки, определенные стандартом языка C++, и библиотеки, подключенные в шаблоне задания.

5 Среда программирования и устройство каркаса

Для упрощения выполнения задания под Windows вам предлагаются файл `cygwin.exe`. Это самораспаковывающийся архив, в нём находится `cygwin` — среда программирования с `gcc`. Разархивируйте `cygwin.exe` и запустите сценарий `Cygwin.bat`. Откроется консоль и вы окажетесь в папке `/home/<username>`. Скопируйте в эту папку архив `gear_project.tar.gz` с каркасом задания и разархивируйте командой `tar xvf gear_project.tar.gz`; зайдите в папку проекта, введя в консоли команду `cd gear_project`. Запустите сценарий компиляции командой `make all`. В папке `build/bin` появятся исполняемые файлы. Их можно запустить из консоли командой `./build/bin/<имя файла>`. Удалить все скомпилированные файлы можно командой `make clean`.

Если вы используете Linux, то вам будет достаточно поставить `gcc` и `GNU make`.

Структура каркаса:

- `src` — директория с исходным кодом в формате `.cpp`. Здесь особенно интересен файл `matrix_example.cpp`, в котором показан пример свертки с `box`-фильтром;
- `include` — директория с заголовочными файлами. Здесь интересны файлы `matrix.h` — объявление класса матрицы и `matrix.hpp` — шаблонная реализация этого класса;
- `externals` — директория с исходными кодами библиотек. Здесь же библиотеки компилируются. В каркасе есть только одна библиотека, с помощью которой загружаются изображения в формате `BM3`;
- `bridge` — директория, в которую добавляются заголовочные файлы и скомпилированные библиотеки для импорта в основной проект;
- `build/bin` — директория, в которой сохраняются выполняемые файлы компиляции (т.е. после того, как отработала команда `make all`);

- `compare.py` — скрипт для проверки программы. Запускается командой `./compare.py <ground_truth.txt> <program_result.txt>`. Скрипт проверяет соответствие разметки полученному результату (Положение центра не должно отличаться более, чем на 3 пикселя, а радиусы более, чем не 2 пикселя) и печатает `FAIL` или `OK`, а также долю шестеренок, для которых верно определено количество зубцов;
- `pic` — папка с картинками;
- `labelling` — папка с разметкой картинок. Разметка описывает индекс правильной шестеренки для замены и параметры всех шестеренок на изображении.

6 Аргументы командной строки

У программы `main`, предназначенной для замены шестеренок, три обязательных аргумента:

- входное изображение;
- выходное изображение;
- текстовый файл для записи параметров найденных объектов.

7 Формат текстового файла

Текстовый файл должен содержать номер шестеренки, которую необходимо установить, и описания всех объектов, найденных на изображении, в формате:

```
<Номер шестеренки>
<Количество найденных объектов>
<Описание первого объекта>
```

...

```
<Описание последнего объекта>
```

Описание каждого объекта начинается с новой строки и имеет следующий вид:

```
<Описание объекта> ::= <Описание оси> | <Описание шестеренки>
```

```
<Описание оси> ::= Axis <Положение центра>
```

```
<Описание шестеренки> ::= Gear <Положение центра> <Внешний радиус> <Внутренний радиус> <Является ли сломанной> <Количество зубцов>
```

```
<Положение центра> ::= <x> <y>
```

```
<Является ли сломанной> ::= TRUE | FALSE
```

Для выполнения базовой части можно для всех шестеренок указывать, что они не сломаны и у них 0 зубцов.

8 Сдача и проверка заданий

Для сдачи задания выполните команду `make clean` в папке проекта, перейдите на папку выше командой `cd ..` и заархивируйте папку с проектом командой `tar cvzf solution.tar.gz gear_project`. Полученный архив `solution.tar.gz` нужно послать в проверяющую систему. Исполняемые программы или скомпилированные библиотеки прикладывать в архив не нужно, сборка будет производится проверяющим в предлагаемой среде программирования.

Проверка заданий будет выполняться на скрытой тестовой выборке, аналогичной выданной. В скрытой выборке 5 простых картинок и 5 картинок со сломанными шестеренками. Для получения баллов за базу необходимо заменить шестеренки на всех простых картинках.