

Описание программной части робота-художника

Латыпов Владимир Витальевич

14 июля 2021 г.

1. Формулировка задачи

Для того, чтобы робот-художник нарисовал что-либо, ему нужно представить данные в определённом формате, а именно — не набор пикселей, как требуется для показа на мониторе, а набор «мазков»: это связано с конструкцией самого робота. Мазки решено было представлять в виде кривых безье второго порядка (то есть квадратичных), к которым добавлены параметры «толщина» и «цвет».

Но на вход подаются рисунки не в векторном, а в растровом формате. Найти такую комбинацию мазков, которая бы лучше всего соответствовала картине/изображению — задача нетривиальная, имеющая множество решений.

Поэтому решено было использовать эвристические алгоритмы оптимизации: Генетический алгоритм и Симуляция отжига.

Функцию ошибки необходимо задать таким образом, чтобы она отражала качество полученной комбинации мазков, причём в любой точке направление её уменьшения соответствовало направлению улучшения результата. Помимо напрашивающегося MSE, используемого

$$MSE = \sum_{y=0}^{y < height} \sum_{x=0}^{x < width} \sum_{c \in \{r,g,b\}} \left(\overrightarrow{\text{rendered}_{x,y,c}} - \overrightarrow{\text{original}_{x,y,c}} \right)^2 \quad (1)$$

2. Общий принцип ГА

Идея работы алгоритма заимствована у природы. Точно также, как в ходе эволюции происходит появление оптимального организма для за-

данных условий, в ходе работы алгоритма ищется набор параметров функции, при котором фитнесс-функция максимальна. В природе

2.1. Термины

Набор параметров представляется в виде «генома» — некой структуры данных, содержащей информацию об этом наборе. В каждый момент времени

3. Авторские Модификации в ГА

Алгоритм реализован мной на языке C++, он хранится в GitHub репозитории <https://github.com/donRumata03/Painter>.

4. Дальнейшее развитие

Несмотря на то, что программа уже работоспособна, есть ещё много идей и планов по её усовершенствованию:

- *Внедрить быстрый пересчёт функции ошибки — это улучшение давно напрашивается, но оно несколько теряет в эффективности из-за того, что в одной мутации в среднем изменяется не так мало мазков (однако это количество убывает со временем). В настоящий момент ведётся работа над внедрением.*
- *Разделение мазков по слоям. Нетрудно заметить, что при рисовании картин художники сначала проходятся по холсту черновыми мазками большого размера, а затем прорабатывают детали. Таких уровней детализации зачастую бывает немало.*

Пример того, как художник (<https://www.youtube.com/watch?v=VaXHtai2alU>) рисует картину по слоям:

Поэтому стоит попробовать сначала заполнять картинку толстыми, грубыми мазками (то есть просто с большей шириной, а в реальной жизни это будет отражаться в большем размере кисти и в более сильном нажатии).

- *Добавить использование локальных методов оптимизации. Такие методы, как градиентный спуск и метод Ньютона позволяют достичь гораздо большей скорости сходимости (в случае метода Ньютона — сходимость квадратичная), но требуют умения посчитать*



Рис. 1:
Фон, ос-
новные
цвета



Рис. 2:
Рельеф у
фона



Рис. 3:
Детализация
объектов
на заднем
плане



Рис. 4:
Основные
объекты

градиент функции ошибки в любой точке, а также вектор вторых производных по каждому из аргументов.

Сами алгоритмы реализованы и находятся в этой папке. Преду-
смотрена опция подсчёта первой и вторых производных через под-
становку близких значений параметров:

$$f'(x_0) \approx \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} \quad (2)$$