**Сравнительный анализ алгоритмов заливки многоугольников.**

Заливка изображений — часто нужная на практике задача, суть которой — заполнить некоторую область изображения, ограниченную контуром, заданным цветом. В данной публикации рассматриваются алгоритмы заливки многоугольников. А также заливка - это алгоритм, определяющий область, «связанную» с определённым элементом в многомерном массиве (как правило, это двумерный массив точек растрового изображения). Алгоритм применяется в графических программах, чтобы определить область, которую следует заполнить определённым цветом.

**Таблица сравнений алгоритмов**

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Достоинства | Недостатки |
| Алгоритм закраски с затравкой | Берётся затравочная точка и закрашивается. Для каждого незакрашенного соседа мы выполняем аналогичную процедуру.  | С помощью этого алгоритма можно закрашивать любые замкнутые области. | Медленный и расходует много памяти для стека |
| Построчная XOR обработка | После того, как границы построены, закрашивание сводится к заполнению в каждой строке промежутков между двумя закрашенными точками. | Преимущество этого алгоритма в его предельной простоте и высокой скорости | Не может работать, при наличии посторонних изображений. |
| Алгоритм XOR для граней с перегородкой | Область инвертируется между ребром и специальной вертикальной линий (т.н. перегородкой). Чаще всего перегородка проводится так, чтобы она пересекала многоугольник | Простота и оригинальность, а также отсутствие дополнительных структур данных | Необходимость выполнения большого числа операций с пикселями (до N операций с каждым пикселем), в том числе и вне многоугольника. В частности, чем больше расстояние между многоугольником и правой границей экрана, тем больше будет совершено "лишних" операций. |

**Заключение**

Рассмотренные алгоритмы заливки многоугольников помогут начинающим программистам быстро разобраться в том, как работает алгоритм и его возможностях, тем самым сократив время на разработку приложений.