

# G Code векторизация след редукция на цветовете за изрисуване с плотер

П. Петров, Г. Костадинов, П. Живков, В. Величкова, Н. Керемедчиева

## G Code Vectorization after Colors Reduction for Plotter Painting

P. Petrov, G. Kostadinov, P. Zhivkov, V. Velichkova, N. Keremedchieva

Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Acad. G. Bonchev St., Bl. 2, 1113 Sofia, Bulgaria, georgi@kostadinov.info

**Key Words:** G Code; vectorization; colors reduction; plotting.

**Abstract.** G Code instructions are used for CNC machines. In order, such machines to produce objects or drawings visual information should be vectorized. Transformation of full-color image to set of 16 colors is achieved by color reduction algorithm. Pixels in the image are replaced with simple shapes (strokes) in such a way that these simple graphic primitives to be drawn by CNC driven 2D plotter. Each stroke should be drawn by a single loading of oil paint. This goal cannot be achieved by standard G Code generation and this research

### 1. Увод

Най-разпространеният стандарт за подаване на инструкции към режещи и плотиращи машини е базиран на множество от инструкции, наречени G Code [1]. Множество CNC машини биват управлявани с G Code инструкции, но обект на това научно изследване е 2D плотер с възможност за нанасяне само на един цвят без подмяна на използваната четка. Целта на изследването е с помощта на плотер да се нанася маслена или акрилна боя върху грундирано платно. Плотерът има възможност да изпълни нанасянето на прости графични примитиви, каквито са отделните черти. За тази цел е необходимо при нанасянето на всяка черта четката да бъде напоявана с маслена или акрилна боя в специално отредена ваничка за тази цел. Тъй като плотерът може да нанася само един цвят без подмяна на ваничката и четката, необходимо е да се организира йерархично нанасяне на различните цветове. Това научно изследване предлага алгоритъм за генериране на нужните G Code инструкции [2], така че нанасянето на маслените или акрилните бои да се осъществи в няколко последователни сесии, които позволяват натрупване на цветовете и изрисуване на цялото изображение с помощта на прост графичен примитив (в случая къса линия или продълговата елипса).

Изложението е организирано в следните отделни секции: секция 2, която разяснява алгоритмите, използвани за редуциране на цветовете и векторизиране на растерната информация; секция 3, която представя алгоритъма за формиране на G Code инструкции; секция 4, даваща

разяснения за някои от проведените експерименти и част от постигнатите резултати; секция 5, която представлява заключение и дава насоки за някои бъдещи разработки.

### 2. Векторизация и редукция на цветовете

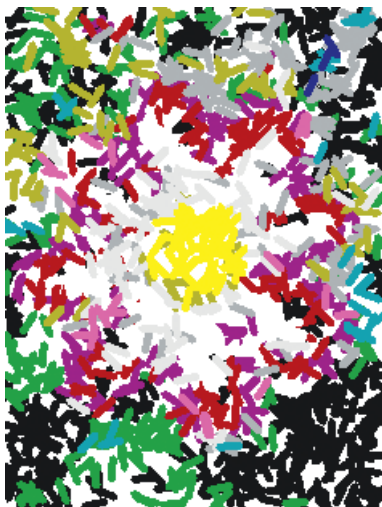
При визуализация някои устройства не позволяват използването на голям набор от различни цветове. За да се визуализира изображение с 16 милиона цвята (пълноцветно растерно изображение) на такива устройства, първо трябва да се редуцират цветовете [3]. При пълноцветните изображения (фиг. 1) за всеки от трите базови цвята (червен, зелен, син) се използват по 8 бита за кодиране на информацията. По този начин се получават малко над 16 милиона различни цвята [4]. Редуцирането на толкова голямо количество цветове до 16 базови цвята е свързано със загуба на огромно количество информация. Редукцията трябва да се направи по такъв начин, че да се запази възможно най-голямо количество от информацията, представена в пълноцветното растерно изображение. Изчислително най-бързият алгоритъм е всеки 16M пиксел да се съпостави на най-близкия до него цвят от 16-цветната палитра. Има множество алтернативи на този подход, като една от тях е използването на евристична оптимизация, която да формира изображение с 16-те базови цвята от основен графичен примитив [5,6].

Като целева функция в евристичните оптимизационни алгоритми се използва евклидово разстояние между двете изображения. Понеже за плотирането има още няколко съществени параметъра, в целевата функция се приема за по-добър резултат, ако апроксимацията е постигната с по-малко на брой графични примитиви. Тъй като не е гарантирано, че графичните примитиви ще покрият изцяло работното поле, то в целевата функция се приема, че по-добрите резултати водят до по-пълно запълване. Наличието на три критерия в целевата функция превръща оптимизационната задача в многокритериална, а това от своя страна налага избор на математическа функция, която би трансформирала задачата в еднокритериална.



Фиг. 1. Пълноцветно растерно изображение

Съществуват множество разработки за апроксимация на изображения с помощта на основни графични примитиви. Най-популярна е апроксимация с триъгълници или неправилни многоъгълници [7]. В това изследване като графичен примитив е избрана разтеглена елипса, която доближава формата на къса линия, нанесена с четка и маслена/акрилна боя (фиг. 2). Графичните примитиви с различен цвят се сортират в намаляващ ред, така че най-често срещаният цвят да се изчертае първи, а най-рядко срещаният цвят да се изчертае последен. По този начин се гарантира, че често срещаните цветове няма изцяло да препокрият рядко срещаните цветове. В групите по цветове е добавен и втори критерий за сортиране (вторичен), който изисква графичните примитиви да са отдалечени максимално един от друг (евклидово разстояние), така че да се даде малко допълнително време за изсъхване на боята, преди да се нанесе следващият графичен примитив. Тази организация на стъпките за изграждане на цялото изображение увеличава с известна степен разминатото разстояние от плотиращата глава.



Фиг. 2. Векторизиране и редукция по цвят

### 3. Трансформация към G code

G Code стандартът е широко използван за подаване на инструкции към режещи и плотиращи машини. По същество представлява група инструкции, които настройват машината за извършване на желаните операции. В това изследване най-често се използват кодовете G0 и G1, които задават преместване на плотиращата глава по осите X, Y и Z. Кодът G0 е за бързо движение, а кодът G1 е за бавно движение. Бързото движение се използва при придвижване на празен ход, докато бавното движение се използва при нанасяне на боята. Преди стартирането на същинското изобразяване с кода G21 се задава използването на милиметри, а с кода G90 се определя използването на абсолютни координати (Листинг 1).

```
gCode += "G21 (All units are in millimeters.);";
gCode += "\n";
gCode += "G90 (Switch to absolute coordinates.);";
gCode += "\n";
gCode += "G00 Z15.00 (Fast pen move up for initialization.);";
gCode += "\n";
gCode += "G00 X0.00 Y0.00 (Fast move to home position for initialization.);";
gCode += "\n";
```

Листинг 1. Мерни единици и координати

За изобразяване на всеки графичен примитив четката първо се изпраща на нулевите координати. След това следва инструкция за потопяване на четката във ваничката с боя. След известен престой четката се вдига и се изпълняват инструкции за придвижване до позицията на графичния примитив. Изпълнява се инструкция за спускане на четката върху грундираното платно и следва инструкция за отместване, така че да се изобрази графичният примитив (Листинг 2).

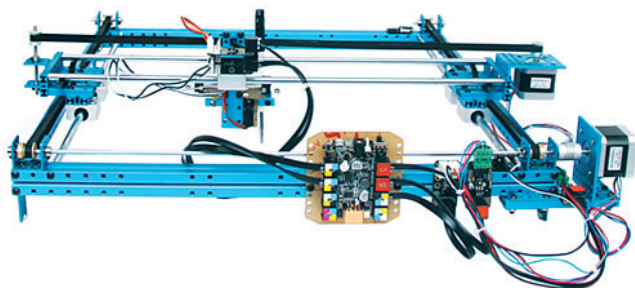
```
String gCode = "G00 Z" + configuration.zUp + " (Fast pen move up.);";
+ "\n" + "G00 X0.00 Y0.00 (Fast move to home position.);";
+ "\n" + "G00 Z" + configuration.zDown + " (Fast pen move down.);";
+ "\n" + "G04 P" + configuration.penRefillTime
+ " (Wait for paint refill before proceeding.);";
+ "\n" + "G00 Z"
+ configuration.zUp + " (Fast pen move up.);";
+ "\n" + "G00 X"
+ Precision.round(
    configuration.xOffset + x1 * configuration.scale, 2)
+ " Y"
+ Precision.round(
    configuration.yOffset + y1 * configuration.scale, 2)
+ " (Fast move to first point position.);";
+ "\n" + "G01 Z"
+ configuration.zDown + " (Slow pen move down.);";
+ "\n"
+ "G01 X"
+ Precision.round(
    configuration.xOffset + x2 * configuration.scale, 2)
+ " Y"
+ Precision.round(
    configuration.yOffset + y1 * configuration.scale, 2)
+ " (Slow move to second point position.);";
+ "\n" + "G01 Z"
+ configuration.zUp + " (Slow pen move up.);";
```

Листинг 2. Изобразяване на примитива

Изобразяването на примитивите се извършва след групиране по цветове, така че за всеки цвят да се генерира отделен комплект инструкции.

## 4. Експерименти и резултати

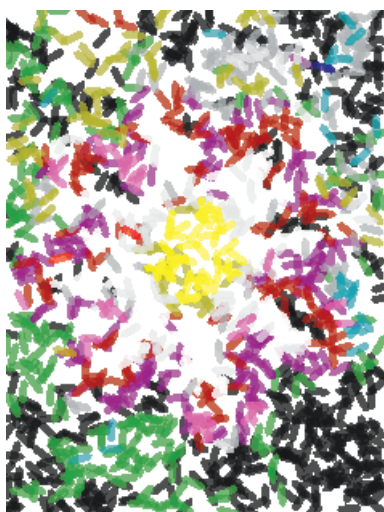
Всички експерименти са проведени с 2Д плотер, позволяващ движение по осите X и Y, както и две позиции (горе/долу) по оста Z (фиг. 3). Значително неудобство представлява нуждата от изсъхване на боята. При работата с акрилни бои изсъхването става по-бързо отколкото при работата с маслени бои. Необходимо е поне частично изсъхване от предходното нанасяне, за да не се получи твърде голямо размазване на предварително нанесените цветове на местата, на които графичните примитиви се препокриват.



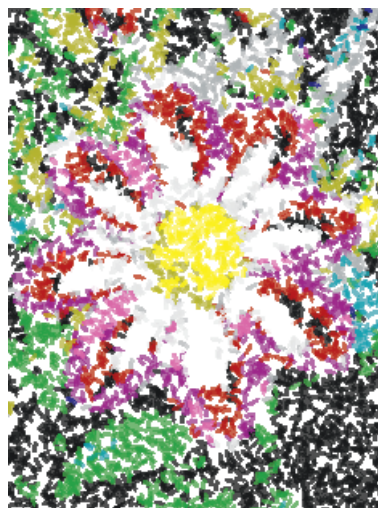
Фиг. 3. Използваният 2Д плотер



Фиг. 4. Апроксимация с голяма четка



Фиг. 5. Апроксимация със средна четка



Фиг. 6. Апроксимация с малка четка

При използване на графичен примитив с различна дължина се получават резултати с различна гъстота на изобразяване (фиг. 4-6).

## 5. Заключение

Представената трансформация към G Code позволява визуализиране на изображения с помощта на векторни примитиви и редуцирани цветове. Предложеното решение е съобразено с факта, че хардуерът не позволява постоянен поток на боя, а е необходимо презареждане при изрисването на всеки отделен графичен примитив.

```

/* Sort ellipses by color with the most used color first. */
List<Ellipse> list = sort();

/*
 * Draw ellipses.
 */
BufferedImage experimental = new BufferedImage(image.getWidth(),
    image.getHeight(), BufferedImage.TYPE_INT_ARGB);
Util.drawEllipses(experimental, list);

/* Multiple-criteria for fitness value estimation. */
double size = list.size();
double distance = Util.distance(image, experimental);
double alpha = Util.alphaLevel(experimental, colors);

return -(1D * size + 100D * distance + 10D * alpha);

```

Листинг 3. Целева функция

В бъдещо изследване би било интересно да се проверят възможностите за боравене с много критерии (Листинг 3). Първото решение, което трябва да се вземе е дали функционалната връзка между критериите да бъде линейна или нелинейна. Ако функционалната връзка е нелинейна, трябва да се избере подходяща форма на нелинейност. И в двата случая остава въпросът в какво съотношение ще

участват критериите при формирането на общата оценка. Тази задача попада в областта на многокритериалният анализ и оптимизация.

Освен графичен примитив елипса е възможно същите експерименти да се повторят с използването на графичен примитив окръжност. В настоящия алгоритъм се използва фиксирана палитра с 16 базови цветове, но като част от оптимизацията може да се въведе и оптимален избор на нюансите за базовите цветове.

## Благодарности

This paper is funded by Velbazhd Software LLC and it is partially supported by the Bulgarian Ministry of Education and Science (contract D01-205/23.11.2018) under the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", approved by DCM # 577/17.08.2018.

Тази статия е финансирана от Велбъжд Софтуер ЕООД и е подкрепена частично от Министерството на образованието и науката (договор Д01-205/23.11.2018 г.) по Националната научна програма „Информационни и комуникационни технологии за единен цифров пазар в науката, образованието и сигурността (ICTinSES)", одобрен от DCM # 577/17.08.2018.

## Литература

1. Patel, P., S. Pavagadhi, S. Acharya. Design and Development of Portable 3-Axis CNC Router Machine. – *International Research Journal of Engineering and Technology*, 6, 2019, No. 3, 1452-1555, ISSN 2395-0072.

2. Aciu, R., H. Ciocarlie, G-code Optimization Algorithm and Its Application on Printed Circuit Board Drilling. Proceedings of 9th IEEE International Symposium on Applied Computational Intelligence and Informatics, Timisoara, 2014, 43-47, ISBN 978-1-4799-4694-5.

3. Papamarkos, N., A. Atsalakis, C. Strouthopoulos. Adaptive Color Reduction. – *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 32, 2002, No. 1, 44-56, ISSN 1083-4419.

4. Kumar, T., K. Verma. A Theory Based on Conversion of RGB Image to Gray Image. – *International Journal of Computer Applications*, 7, 2010, No. 2, 7-10, ISSN 0975-8887.

5. Evtimov, G., D. Keremedchiev, M. Barova. Image Vectorization and Colors Reduction with Ant Colony Optimization, Image Vectorization and Colors Reduction with Ant Colony Optimization. – *Advances in Mathematics: Scientific Journal*, 5, 2016, No. 2, 153-160, ISSN 1857-8365.

6. Balabanov, T., M. Barova, D. Keremedchiev. Image Construction with 2D Ellipses by Genetic Algorithms Optimization. Proceedings of 11th Annual Meeting of the Bulgarian Section of SIAM, Fastumprint, 2016, 10-11, ISSN 1313-3357.

7. Johansson, R. Genetic Programming: Evolution of Mona Lisa. <https://rogerjohansson.blog/2008/12/07/genetic-programming-evolution-of-mona-lisa/>, 2008.

За контакти:

Докторант инж. **Георги Костадинов**  
Институт по информационни  
и комуникационни технологии – БАН  
e-mail: georgi@kostadinov.info