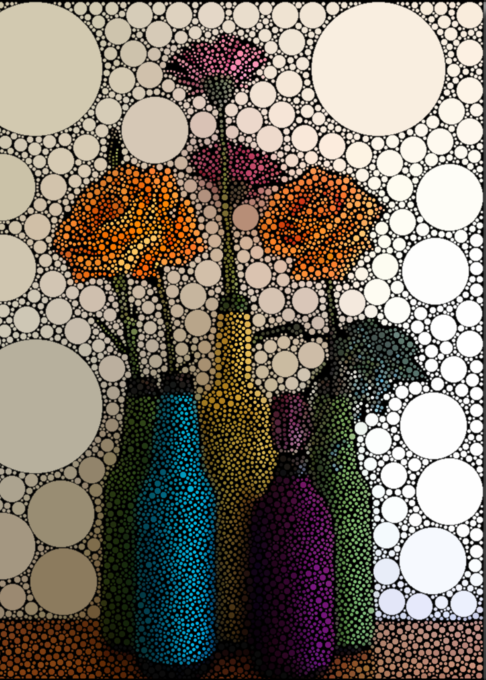


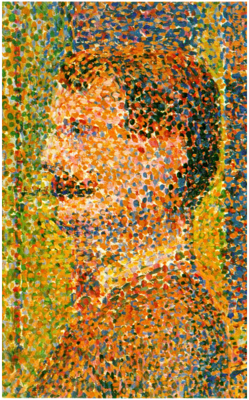
Digital Circlism

as Algorithmic Art

S. De
P. Bhowmick

Background

According to artists and critics, it's a fusion of *pop art* and *pointillism*[a]



[a] Chipp, H.B. 1996: *Theories of Modern Art*

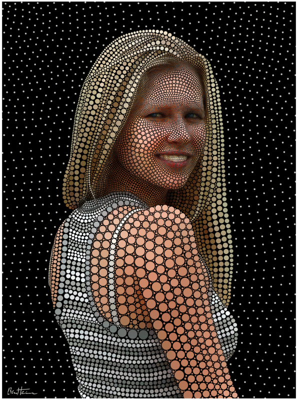
Background

First introduced by *Edward C. Stresino* in 1985[a]

[a] http://www.circlism.com

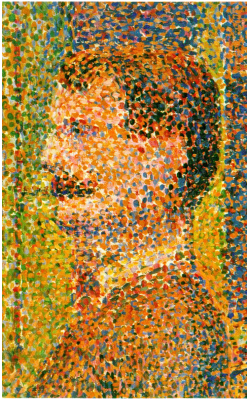
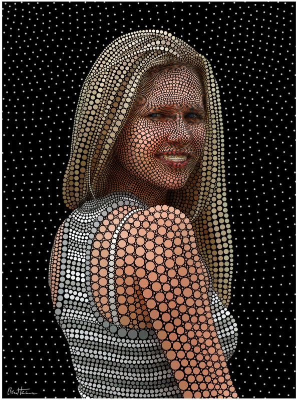
Background

Digital artworks by
Ben Heine
since 2010[a]



[a] http://benheine.deviantart.com/gallery/30782139, http://www.flickr.com/photos/benheine/sets/72157623553428960

Background

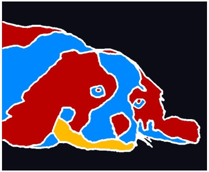
Pointillism	Pop art	Digital circlism
		
Georges Seurat (France, 1859-1891)	*Romero Britto* (Brazil, b. 1963)	*Ben Heine* (Ivory Coast, b. 1983)

Our Algorithm

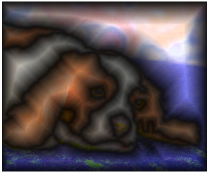
Basic Steps

input

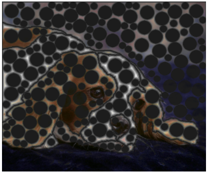
segmented



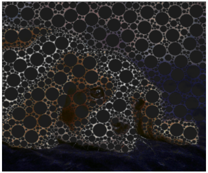
parsed



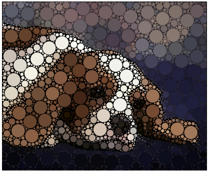
EDT



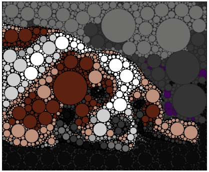
$r = 20, 10$



$r = 20, 10, 5, 2$

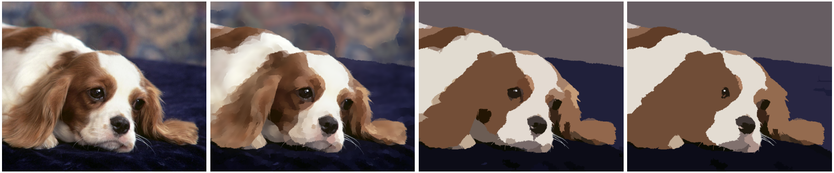


coloring



Macbeth Luv

Mean shift segmentation



Input $(h_s, h_r) = (7, 6)$ $(11, 10)$ $(14, 13)$

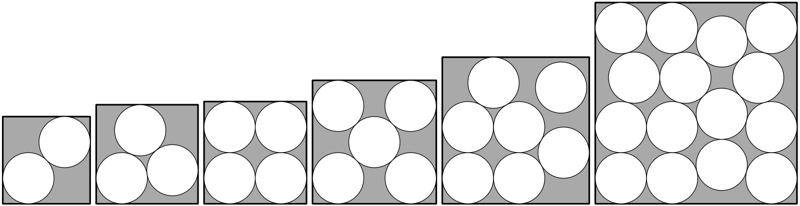
Two types of bandwidth parameters:[a,b]

- h_s in spatial domain $(d = 2)$
- h_r in CIELUV color subspace $(d = 3)$

[a] Comaniciu & Meer: Mean shift - A robust approach toward feature space analysis. IEEE PAMI (2002)

[b] Fairchild: Color Appearance Models. Addison-Wesley (1998)

Circle packing (1)



2	3	4	5	7	15
$\rho = 0.53901$	0.60965	0.78540	0.67377	0.66931	0.76206

Packing density

$\rho =$ proportion of the region covered by packing circles—forms the *maximization criterion*.

Circle packing

Solution to date: Only for packing inside geometric primitives like *squares*, *circles*, *triangles*, etc.

Difficulty: Many packing problems are *NP-hard*.[c,d]

Our work: Packing circles in *discrete space* instead of real space, where the circles are defined by a *finite and small set of radii*.

[c] Demaine *et al.*: Circle packing for origami design is hard. arxiv.org (2010)

[d] Melissen: Packing 16, 17 or 18 circles in an equilateral triangle. Discrete Mathematics (1995)

Circle packing

Solution to date: Only for packing inside geometric primitives like *squares, circles, triangles,* etc.

Difficulty: Many packing problems are *NP-hard*.[c,d]

Our work: Packing circles in *discrete space* instead of real space, where the circles are defined by a *finite and small set of radii.*

[c] Demaine *et al.*: Circle packing for origami design is hard. arxiv.org (2010)

[d] Melissen: Packing 16, 17 or 18 circles in an equilateral triangle. Discrete Mathematics (1995)

Circle packing (2)

Solution to date: Only for packing inside geometric primitives like *squares, circles, triangles,* etc.

Difficulty: Many packing problems are *NP-hard*.[c,d]

Our work: Packing circles in *discrete space* instead of real space, where the circles are defined by a *finite and small set of radii*.

---

[c] Demaine *et al.*: Circle packing for origami design is hard. arxiv.org (2010)

[d] Melissen: Packing 16, 17 or 18 circles in an equilateral triangle. Discrete Mathematics (1995)

Circle packing (3)

Random space filling[e]**:** Attempts to solve the problem by iterative filling.

Problems

- Increasing inefficiency as more and more circles are packed.
- Situation worsens in case of region with an arbitrary shape—a usual outcome of natural object segmentation.

[e] http://paulbourke.net/texture colour/randomtile.

Circle packing (3)

Random space filling[e]**:** Attempts to solve the problem by iterative filling.

Problems

- Increasing inefficiency as more and more circles are packed.
- Situation worsens in case of region with an arbitrary shape—a usual outcome of natural object segmentation.

[e] http://paulbourke.net/texture colour/randomtile.

Circle packing (3)

Random space filling[e]**:** Attempts to solve the problem by iterative filling.

Problems

- Increasing inefficiency as more and more circles are packed.
- Situation worsens in case of region with an arbitrary shape—a usual outcome of natural object segmentation.

[e] http://paulbourke.net/texture colour/randomtile.

Circle packing

Our solution

- Compute *Euclidean distance transform (EDT)*
- Use EDT to pack *denomination circles* in different segments.
- *DP technique* of solving the coin denomination problem.[f]
- *Recompute the EDT* of a segment after packing a circle and then greedily check feasibility of placing the next circle.

---

[f] Shallit: What this country needs is an 18c piece. Mathematical Intelligencer (2003).

Circle packing

Our solution

- Compute *Euclidean distance transform (EDT)*
- Use EDT to pack *denomination circles* in different segments.
- *DP technique* of solving the coin denomination problem.[f]
- *Recompute the EDT* of a segment after packing a circle and then greedily check feasibility of placing the next circle.

[f] Shallit: What this country needs is an 18c piece. Mathematical Intelligencer (2003).

Circle packing

Our solution

- Compute *Euclidean distance transform (EDT)*
- Use EDT to pack *denomination circles* in different segments.
- *DP technique* of solving the coin denomination problem.[f]
- *Recompute the EDT* of a segment after packing a circle and then greedily check feasibility of placing the next circle.

[f] Shallit: What this country needs is an 18c piece. Mathematical Intelligencer (2003).

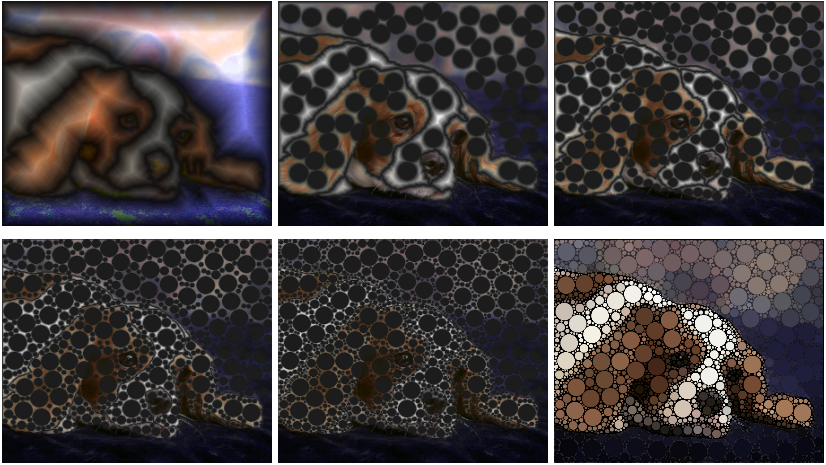
Circle packing

Our solution

- Compute *Euclidean distance transform (EDT)*
- Use EDT to pack *denomination circles* in different segments.
- *DP technique* of solving the coin denomination problem.[f]
- *Recompute the EDT* of a segment after packing a circle and then greedily check feasibility of placing the next circle.

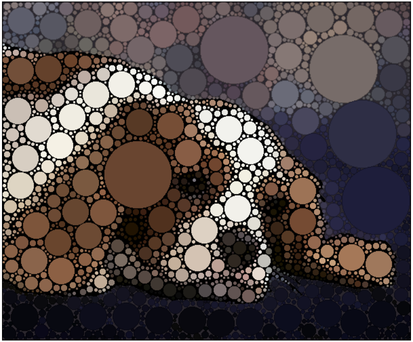
[f] `Shallit: What this country needs is an 18c piece. Mathematical Intelligencer (2003).`

Circle packing

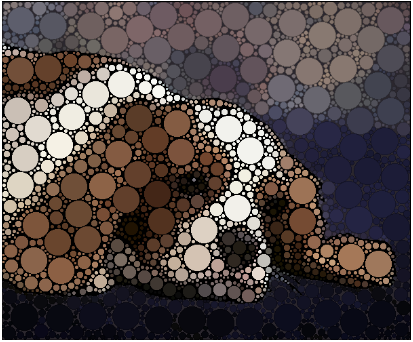


denomination set $D_2 = \{20, 10, 5, 3, 2\}$

Circle packing



$D_1 = \{\mathbf{200}, \mathbf{100}, \mathbf{50}, 20, 10, 5, 3, 2\}$



$D_3 = \{5, 3, 2\}$

Runtime

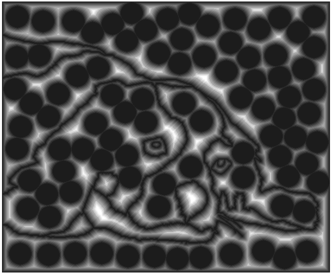
EDT → $O(n)$ *time*, $n = \#$ image pixels.

For packing circles of radius r

Local maxima finding
→ $O(n)$ *time.*

EDT recomputation in R after packing each circle
→ $O(|R|)$ time in *worst case.*

Let $c_R = \#$circles packed in R.
$\pi r^2 c_R \leqslant |R| \Rightarrow c_R = O(|R|/r^2)$.

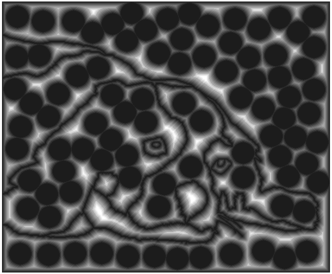


Runtime (1)

EDT $\rightarrow$ $O(n)$ *time*, $n = \#$ image pixels.

For packing circles of radius r

Local maxima finding
$\rightarrow$ $O(n)$ *time*.



EDT recomputation in R after packing each circle
$\rightarrow$ $O(|R|)$ time in *worst case*.

Let $c_R = \#$circles packed in R.
$\pi r^2 c_R \leqslant |R| \Rightarrow c_R = O(|R|/r^2)$.

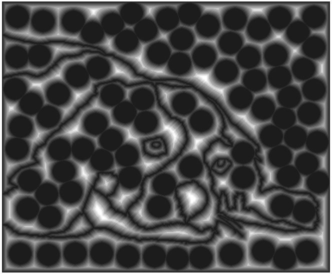
Runtime

EDT → $O(n)$ *time*, $n = \#$ image pixels.

For packing circles of radius r

Local maxima finding
→ $O(n)$ *time*.

EDT recomputation in R after packing each circle
→ $O(|R|)$ time in *worst case*.



Let $c_R = \#$circles packed in R.
$\pi r^2 c_R \leqslant |R| \Rightarrow c_R = O(|R|/r^2)$.

Runtime

(1)

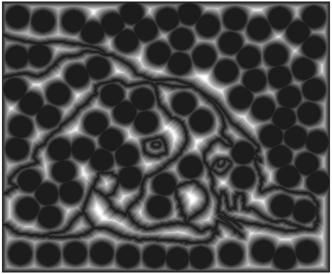
EDT → $O(n)$ *time*, $n = \#$ image pixels.

For packing circles of radius r

Local maxima finding
→ $O(n)$ *time*.

EDT recomputation in R after packing each circle
→ $O(|R|)$ time in *worst case*.

Let $c_R = \#$circles packed in R.
$\pi r^2 c_R \leqslant |R| \Rightarrow c_R = O(|R|/r^2)$.



Runtime (2)

Thus, to compute EDT over all the r-radius circles packed in R, worst-case time complexity is

$$\begin{aligned} T(r) &= \sum_{R \subset I} c_R |R|, \text{ where } c_R = O(|R|/r^2) \\ &= \sum_{R \subset I} O\left(\left(\frac{|R|}{r} \right)^2 \right) \\ &= O\left(\sum_{R \subset I} \left(\frac{|R|}{r} \right)^2 \right) \\ &= O\left(\frac{n^2}{r^2} \right). \end{aligned}$$

Runtime

Thus, to compute EDT over all the r-radius circles packed in R, worst-case time complexity is

$$
\begin{aligned}
T(r) &= \sum_{R \subset I} c_R |R|, \text{ where } c_R = O(|R|/r^2) \\
&= \sum_{R \subset I} O\left(\left(\frac{|R|}{r}\right)^2\right) \\
&= O\left(\sum_{R \subset I} \left(\frac{|R|}{r}\right)^2\right) \\
&= O\left(\frac{n^2}{r^2}\right).
\end{aligned}
$$

Runtime (2)

Thus, to compute EDT over all the r-radius circles packed in R, worst-case time complexity is

$$
\begin{aligned}
T(r) &= \sum_{R \subset I} c_R |R|, \text{ where } c_R = O(|R|/r^2) \\
&= \sum_{R \subset I} O\left(\left(\frac{|R|}{r} \right)^2 \right) \\
&= O\left(\sum_{R \subset I} \left(\frac{|R|}{r} \right)^2 \right) \\
&= O\left(\frac{n^2}{r^2} \right).
\end{aligned}
$$

Runtime

(2)

Thus, to compute EDT over all the r-radius circles packed in R, worst-case time complexity is

$$\begin{aligned} T(r) &= \sum_{R\subset I} c_R|R|, \text{ where } c_R = O(|R|/r^2) \\ &= \sum_{R\subset I} O\left(\left(\frac{|R|}{r}\right)^2\right) \\ &= O\left(\sum_{R\subset I}\left(\frac{|R|}{r}\right)^2\right) \\ &= O\left(\frac{n^2}{r^2}\right). \end{aligned}$$

Runtime (2)

Thus, to compute EDT over all the r-radius circles packed in R, worst-case time complexity is

$$\begin{aligned} T(r) &= \sum_{R \subset I} c_R |R|, \text{ where } c_R = O(|R|/r^2) \\ &= \sum_{R \subset I} O\left(\left(\frac{|R|}{r}\right)^2\right) \\ &= O\left(\sum_{R \subset I} \left(\frac{|R|}{r}\right)^2\right) \\ &= O\left(\frac{n^2}{r^2}\right). \end{aligned}$$

Runtime

(3)

If $|D| = k$, then total circle packing runtime is

$$T = \sum_{r \in D} T(r) = O\left(\frac{kn^2}{r_k^2}\right),$$

where $r_k = \min\{r_i : 1 \leqslant i \leqslant r_k\}$.

Note: We've not considered the decreasing nature of the effective packing area of a region R as circles are packed into it. So, $O(|R|)$ runtime for every recomputation of EDT is quite a loose bound.
A probabilistic analysis can possibly bring down the average time complexity.

Actual CPU time: Runtime increases with increase in

Runtime

If $|D| = k$, then total circle packing runtime is

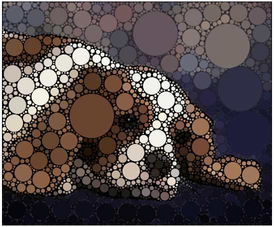
$$T = \sum_{r \in D} T(r) = O\left(\frac{kn^2}{r_k^2}\right),$$

where $r_k = \min\{r_i : 1 \leqslant i \leqslant r_k\}$.

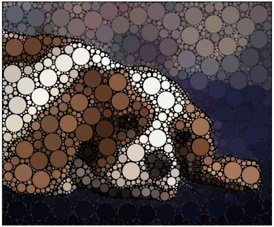
Note: We've not considered the decreasing nature of the effective packing area of a region R as circles are packed into it. So, $O(|R|)$ runtime for every recomputation of EDT is quite a loose bound.
A probabilistic analysis can possibly bring down the average time complexity.

Actual CPU time: Runtime increases with increase in

Runtime



$|D_1| = 8$
69 secs.
$\rho = 0.827$

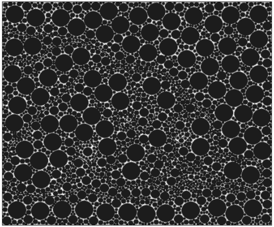


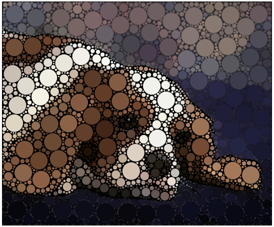
$|D_2| = 5$
18 secs.
$\rho = 0.824$

$|D_3| = 3$
16 secs.
$\rho = 0.749$

Color Rendition

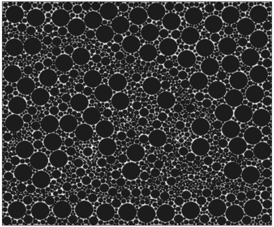
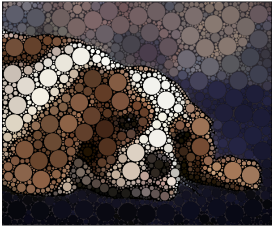
(1)





- Choose the primer colors for foreground and background.
- Color each circle by its average color from the original image.

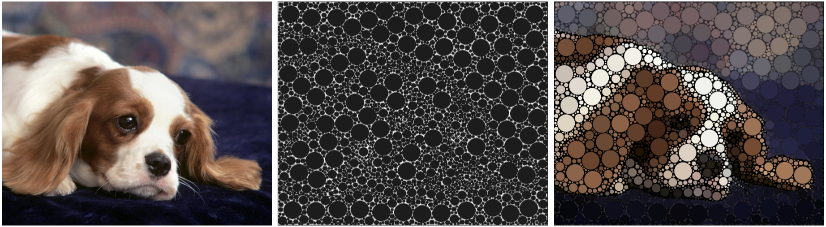
Color Rendition

- Choose the primer colors for foreground and background.
- Color each circle by its average color from the original image.

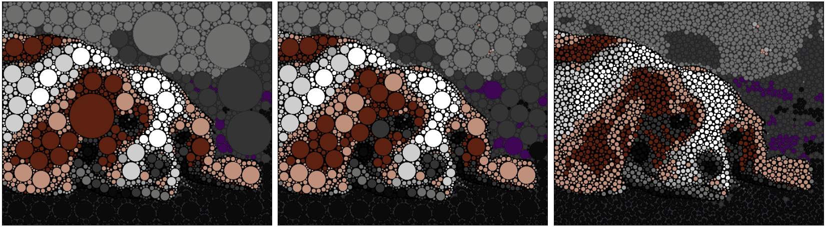
Color Rendition

(1)



- Choose the primer colors for foreground and background.
- Color each circle by its average color from the original image.

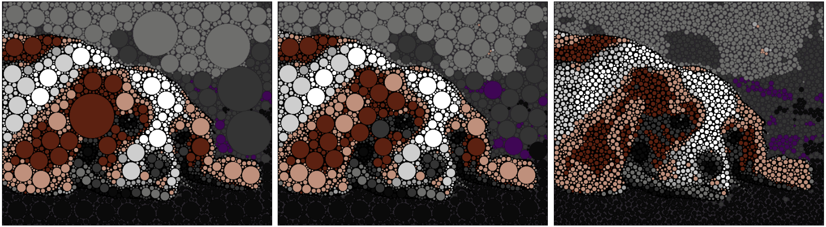
Color Rendition



Macbeth Color Mapping

- Replace the circle color by the nearest color of the Macbeth chart.
- Nearness can be measured in different color subspaces (RGB, CIELUV, ...).

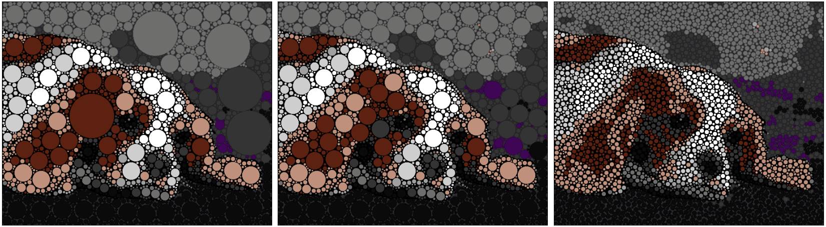
Color Rendition



Macbeth Color Mapping

- Replace the circle color by the nearest color of the Macbeth chart.
- Nearness can be measured in different color subspaces (RGB, CIELUV, ...).

Color Rendition



Macbeth Color Mapping

- Replace the circle color by the nearest color of the Macbeth chart.
- Nearness can be measured in different color subspaces (RGB, CIELUV, ...).

Color Rendition

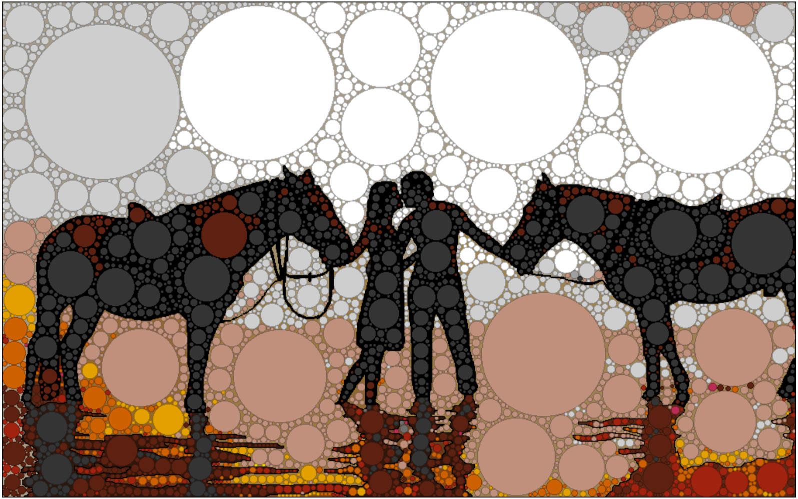
Color Rendition

(3)

No mapping

$\rho = 0.904$, *2 mins. 14 secs.*

Color Rendition

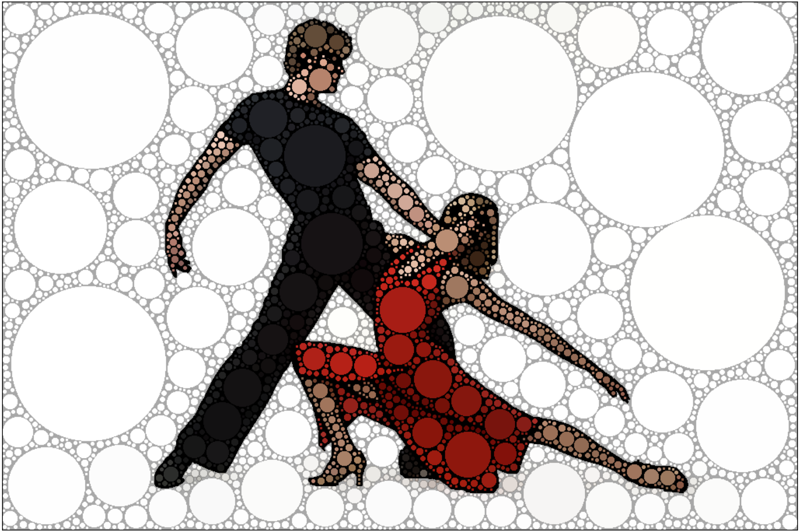


Macbeth color mapping to nearest

Color Rendition

Macbeth color mapping to 2nd nearest

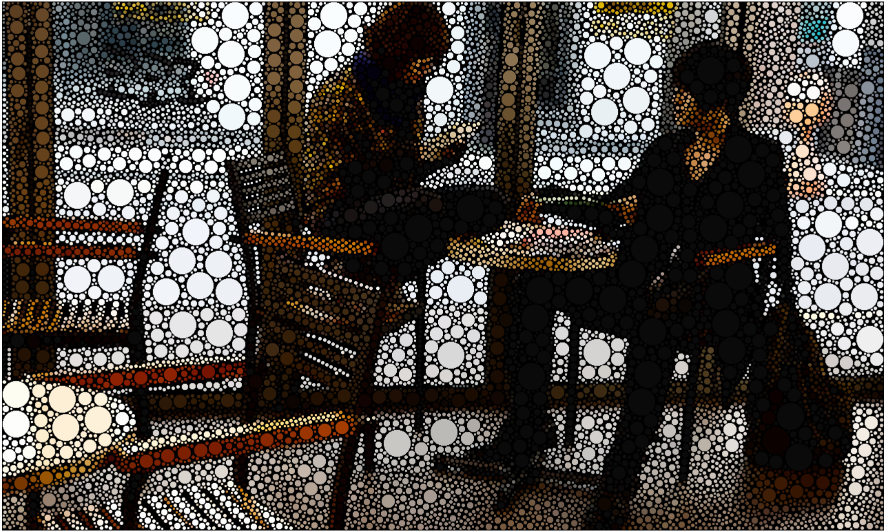
Artwork Results



Artwork Results

Artwork Results

Artwork Results

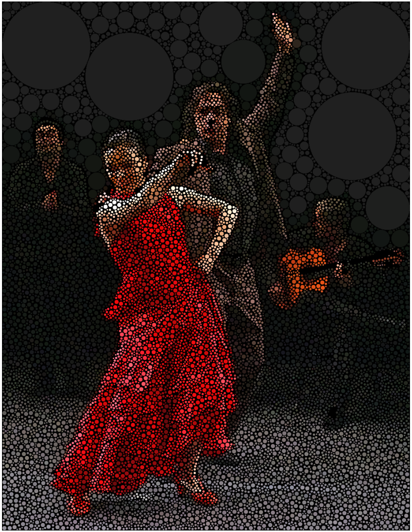


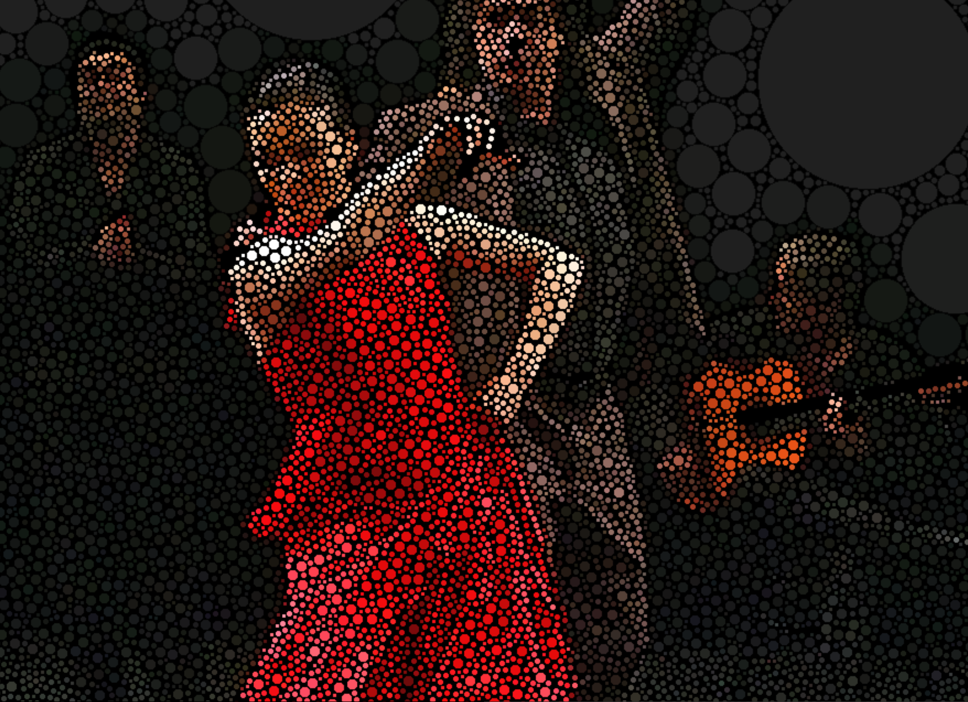
Artwork Results

Artwork Results

Artwork Results

Artwork Results






Thank You