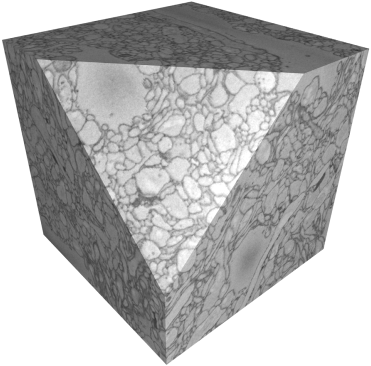
Computer Vision I

Bjoern Andres, Holger Heidrich, Jannik Presberger

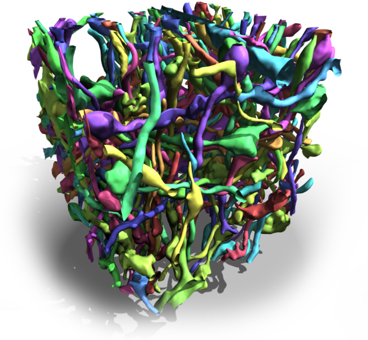
Machine Learning for Computer Vision
TU Dresden

Winter Term 2023/2024

- So far, we have studied **pixel classification**, a problem whose feasible solutions define decisions at the pixels of an image
- Next, we will study **image decomposition**, a problem whose feasible solutions decide whether pairs of pixels are assigned to the same or distinct components of the image
- Image decomposition has applications where components of the image are indistinguishable by appearance (see next slide)

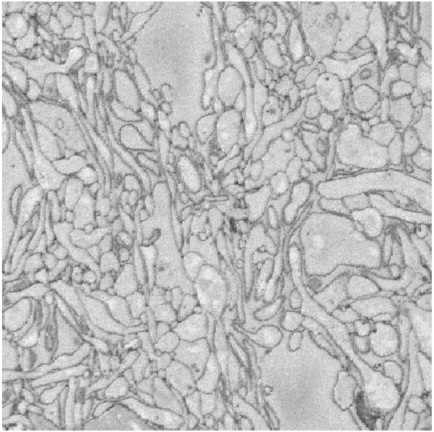


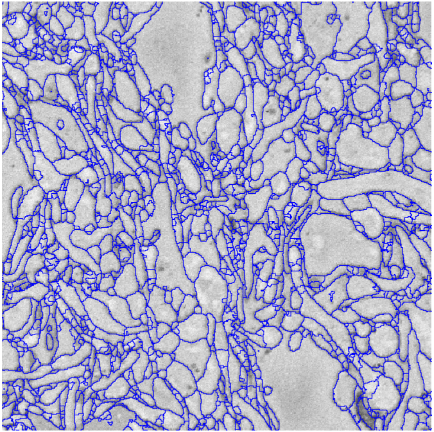
$\mapsto$

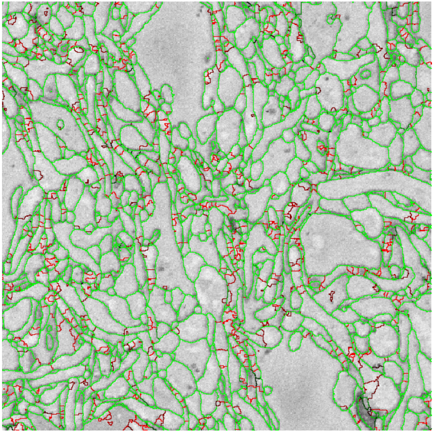


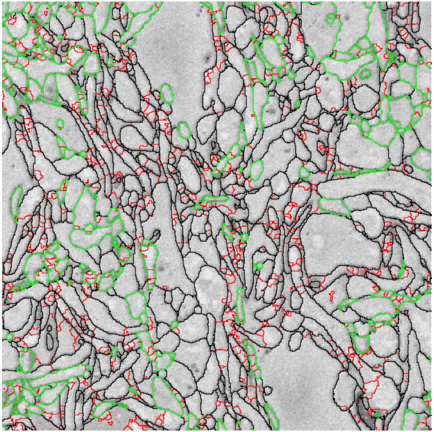
Volume Image (32 nm/voxel)
(Denk and Horstmann, 2004)

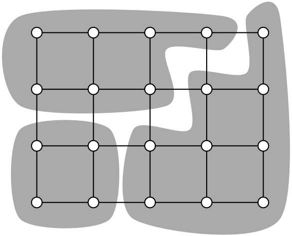
Decomposition
(Andres et al., 2012)





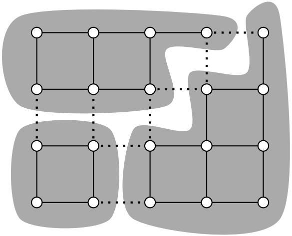






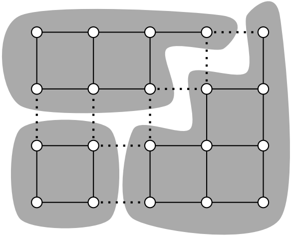
Decomposition of a graph $G = (V, E)$

- A mathematical abstraction of a decomposition of an image is a decomposition of the pixel grid graph.
- A decomposition of a graph is a partition of the node set into connected subsets (one example is depicted above in gray).



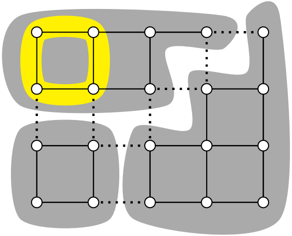
Decomposition of a graph $G = (V, E)$

- A decomposition of a graph is characterized by the set of edges that straddle distinct components (depicted above as dotted lines)
- Those subsets of edges are called **multicuts** of the graph



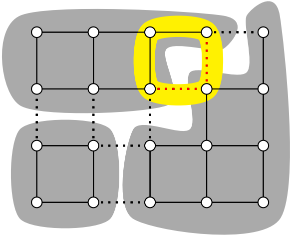
Multicut of a graph $G = (V, E)$

- A decomposition of a graph is characterized by the set of edges that straddle distinct components (depicted above as dotted lines)
- Those subsets of edges are called **multicuts** of the graph



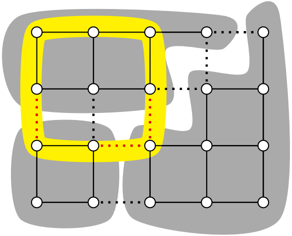
Multicut of a graph $G = (V, E)$

- The defining property of multicuts is that no cycle in the graph intersects with the multicut in precisely one edge



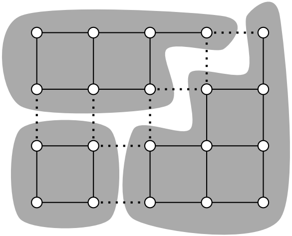
Multicut of a graph $G = (V, E)$

- The defining property of multicuts is that no cycle in the graph intersects with the multicut in precisely one edge



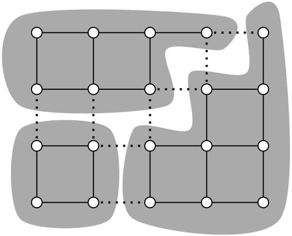
Multicut of a graph $G = (V, E)$

- The defining property of multicuts is that no cycle in the graph intersects with the multicut in precisely one edge

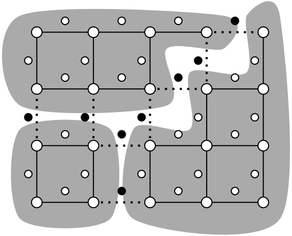


Multicut of a graph $G = (V, E)$

$$\text{multicuts}(G) := \{M \subseteq E \,|\, \forall C \in \text{cycles}(G) : \, |M \cap C| \neq 1\}$$

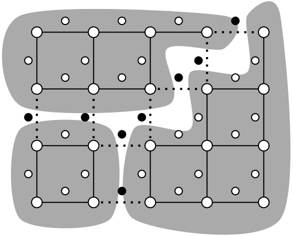


Multicut of a graph $G = (V, E)$



Multicut of a graph $G = (V, E)$

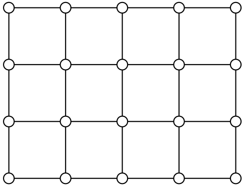
- The characteristic function $y\colon E \to \{0, 1\}$ of a multicut $y^{-1}(1)$ can be used to encode the decomposition induced by the multicut in an $|E|$-dimensional 01-vector
- For any $e \in E$, $y_e = 1$ indicates that an edge is cut, straddling distinct components



Multicut of a graph $G = (V, E)$

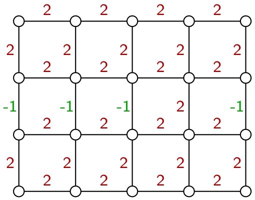
- The set of the characteristic functions of all multicuts of G:

$$Y_G := \left\{ y : E \to \{0,1\} \;\middle|\; \forall C \in \mathsf{cycles}(G)\, \forall e \in C :\, y_e \leq \sum_{f \in C \setminus \{e\}} y_f \right\}$$



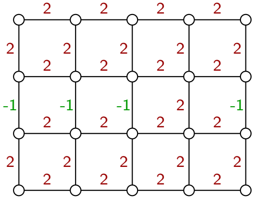
Graph $G = (V, E)$

- An instance of the image decomposition problem is given by a graph $G = (V, E)$ and, for every edge $e = \{v, w\} \in E$, a (positive or negative) cost $c_e \in \mathbb{R}$ that is payed iff the incident pixels v and w are put in distinct components
- Such costs are often estimated from examples using machine learning technqiues



Graph $G = (V, E)$. Edge costs $c : E \to \mathbb{R}$

- An instance of the image decomposition problem is given by a graph $G = (V, E)$ and, for every edge $e = \{v, w\} \in E$, a (positive or negative) cost $c_e \in \mathbb{R}$ that is payed iff the incident pixels v and w are put in distinct components
- Such costs are often estimated from examples using machine learning technqiues

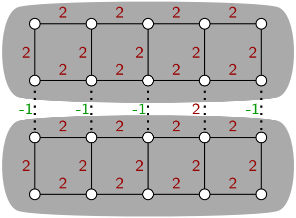


Graph $G = (V, E)$. Edge costs $c : E \to \mathbb{R}$

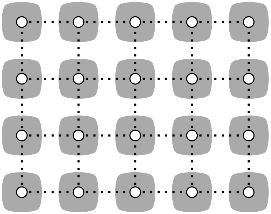
- Image decomposition problem:

$$\min_{y \in Y_G} \sum_{e \in E} c_e \, y_e$$

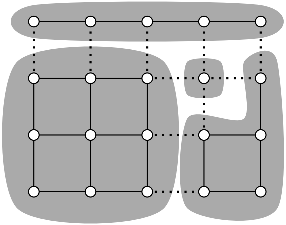
- The optimal solution is shown in the next slide



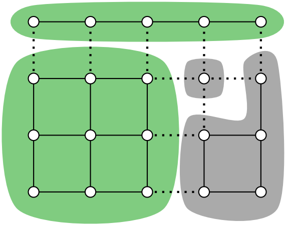
Graph $G = (V, E)$. Edge costs $c : E \to \mathbb{R}$



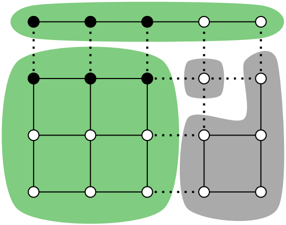
- One technique for finding feasible solutions to an image decomposition problem is **local search**.
- Starting from the finest decomposition into singleton components (depicted above), we greedily join neighboring components as long as this improves the cost (see next slide).



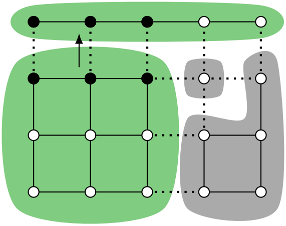
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components



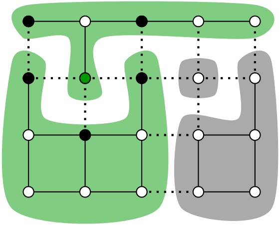
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green)



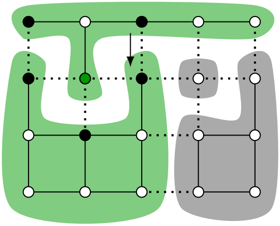
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black)



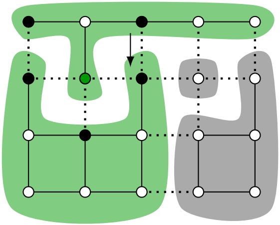
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.



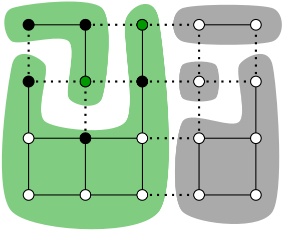
- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost



- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost



- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost



- Once no joining of neighboring components further reduces the cost, we consider all pairs of neighboring components (depicted in green) and all nodes at the shared boundary (depicted in black) and all possibilities of moving nodes from one component to the other.
- The procedure is iterated until no such transformation further reduces the cost