Exercise Sheet 2
Room: 02.09.023
Tue, 29.04.2014, 14:00-15:00
Submission deadline: Mon, 28.04.2014, 23:59 to windheus@in.tum.de

Mathematics: Metric Spaces

Exercise 1 (From the lecture, One point). Show that...

1. ... every isometry is an homeomorphism.
2. ... every isometry is injective.
3. ... being isometric is an equivalence relation.
4. ... dist$_X$(x, S) is a nonexpanding function.

Let f : X → Y, g : Y → Z be bi-Lipschitz homeomorphisms. Show that...

5. ... g ◦ f : X → Z is a bi-Lipschitz homomorphism.
6. ... dil(g ◦ f) ≤ dil(f) · dil(g)
7. ... the Lipschitz distance is a metric.

Exercise 2 (Continuity, One point). Let X ⊂ R$^2$, Y ⊂ R$^3$. Find metrics $d_X$, $d_Y$, such that...

1. ... every function f : X → Y is continuous.
2. ... the only continuous functions f : X → Y are the constant functions.

Show that...

3. ... every Lipschitz function is continuous.
4. ... there exist continuous functions that are not Lipschitz.
Programming: Farthest Point Sampling

Exercise 3 (One point). Download and expand the file `exercise2.zip` from the lecture website. Modify the files `euclideanfps.m`, `euclideanvoronoi.m` to implement the functions as explained below. You can run the script `exercise.m` to test and visualize your solutions.

1. Given a triangle mesh \( \mathcal{M} = (\mathcal{V}, \mathcal{F}) \), a metric \( d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0} \), a number \( K \in \mathbb{N} \) and an initial vertex \( v_1 \in \mathcal{V} \), the farthest-point-sampling method computes a set of \( K \) vertices by the following algorithm:

\[
\text{for } i = 2 \text{ to } K \text{ do} \\
v_i \leftarrow \arg \max_{v \in \mathcal{V}} (\min_{j \in \{1, \ldots, i-1\}} d(v, v_j)) \\
\text{end for} \\
\text{return } \{v_1, \ldots, v_K\}
\]

Implement function `euclideanfps` that computes the farthest point sampling of a mesh with respect to the euclidean metric. The resulting set of vertices should be returned as a vector of indices.

2. Given a triangle mesh \( \mathcal{M} = (\mathcal{V}, \mathcal{F}) \) and a metric \( d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0} \), the Voronoi cells with respect to a set of \( K \in \mathbb{N} \) vertices \( \{v_1, \ldots, v_K\} \subset \mathcal{V} \) are defined as the partition \( \mathcal{V}_1 \cup \ldots \cup \mathcal{V}_K \cup \mathcal{B} = \mathcal{V} \) such that

\[
v \in \mathcal{V}_i \iff (\forall j \neq i : d(v, v_i) < d(v, v_j)),
\]

for all \( i \in \{1, \ldots, K\}, v \in \mathcal{V} \).

If we assume \( \mathcal{B} = \emptyset \) the partition into Voronoi cells can be represented by a function \( f : \mathcal{V} \to \{1, \ldots, K\} \). Implement function `euclideanvoronoi` that computes this partition of vertices into Voronoi cells with respect to the euclidean metric. (If you encounter vertices lying on the boundary \( \mathcal{B} \), just pick the index of one of the neighbouring cells.)

Exercise 4 (One point). Modify the files `metricfps.m`, `metricvoronoi.m`, `distortion.m` and `ghdistance.m` to implement the functions as explained below. You can run the script `exercise.m` to test and visualize your solutions.

1. Implement function `metricfps` that computes the farthest point sampling of a mesh with respect to a metric given by matrix \( D \in \mathbb{R}^{||\mathcal{V}|| \times ||\mathcal{V}||} \). The resulting set of vertices should be returned as a vector of indices.

2. Implement function `metricvoronoi` that computes the partition of vertices into Voronoi cells with respect to a metric given by matrix \( D \in \mathbb{R}^{||\mathcal{V}|| \times ||\mathcal{V}||} \).

3. Now we want to match two shapes, i.e. two set of vertices \( \mathcal{V}_1, \mathcal{V}_2 \), both equipped with a metric represented by \( D_1 \in \mathbb{R}^{||\mathcal{V}_1|| \times ||\mathcal{V}_1||}, D_2 \in \mathbb{R}^{||\mathcal{V}_2|| \times ||\mathcal{V}_2||} \). To make the task computationally tractable we will use farthest point sampling to obtain two smaller subsets \( S_1 \subset \mathcal{V}_1, S_2 \subset \mathcal{V}_2 \), such that \( |S_1| = |S_2| \). We will represent
the metric $D_1$ restricted to $S_1$ by matrix $\hat{D}_1 \in \mathbb{R}^{|S_1| \times |S_1|}$. Analogically define $\hat{D}_2 \in \mathbb{R}^{|S_2| \times |S_2|}$. A surjective correspondence can be represented by function $f : \{1, \ldots, |S_1|\} \to \{1, \ldots, |S_2|\}$, where $f(i) = j$ means that vertex $v_i \in S_1$ is set into correspondence with $v_j \in S_2$.

Implement function distortion that computes the distortion of given function $f : \{1, \ldots, |S_1|\} \to \{1, \ldots, |S_2|\}$ with respect to metrics $\hat{D}_1 \in \mathbb{R}^{|S_1| \times |S_1|}, \hat{D}_2 \in \mathbb{R}^{|S_2| \times |S_2|}$. Recall that the distortion is defined by

$$\text{dis}(f) = \sup_{v, w \in S_1} |\hat{D}_1(v, w) - \hat{D}_2(f(v), f(w))|.$$ 

4. Implement function ghdistance that computes the Gromov-Hausdorff distance between the two sets $S_1, S_2$ and also returns the optimal permutation $f^*$. You can use exhaustive search over the space of all functions $f : \{1, \ldots, |S_1|\} \to \{1, \ldots, |S_2|\}$. 
