Description of Stability Analysis Process and Results

Document UUID: d0b509df-70a7-4356-8b68-7b083d8b5208

David Bayani [redacted]

day29 month5 year2019

1 Intended Receiptiants

David Held [redacted]
Stefan Mitsch [redacted]
The author only provides permission for the above two listed people to view and read this document.

2 Overview

On my own accord, I examined the stability of trained policies that Edward Ahn had provided [1] [2] . In particular, if $\pi : \mathbb{R}^k \to \mathbb{R}^n$ is a policy, I examined bounds on $\widetilde{\pi}(y) = \textbf{measure}(\{\pi(x) | x \in y + [-\epsilon, \epsilon]^k\})$, where y is an arbitrary given member of the input space $\mathbb{R}^k$, ϵ is a positive real bounding perturbations to the input, and **measure** is some defined method of assigning a size to a set (a measure of the set). This is a form of analysis using abstract interpretation domains Cousot and Cousot (1977) where the domain of abstraction are boxes. Boxes are convienient since they are easy to define, and we can find vectors in such sets by considering each coordinate independant of the other coordinate values; this latter point is in contrast to other convex polytopes such as, say, a pyrimid, where the values that one variable may take-on is a function of the assigned values of the other variables. Note that this independance-of-coordinates makes linear optimization over boxes trivial, a fact we use below. While boxes are convienient computationally and implementationally, they are not necessarly the smallest set that contains realistic perturbations of input states, and thus provide a looser bound over alternatives. As a seperate note, observe that the process we will describe is related to local reachability analysis, since we compute the image of $y + [-\epsilon, \epsilon]^k$ under π.

In the description thus far, there is a presumption of a given, fixed y around which we want to examine the behavior of π. In order to examine the general stability of π over its *properly bounded* input space, $\mathscr{I} \subseteq \mathbb{R}^k$, we take y as uniform samples from $\mathscr{I}$ and set ϵ to be large enough to componesate for gaps between samples, which allows us to obtain a bound on the stability [3] for each element of $\mathscr{I}$ for perturbations bounded by a smaller amount ϵ', $0 < \epsilon' < \epsilon$. For instance, if $\mathscr{I} = [-1, 1)$ and we are interested in the stability of π under perturbations up to magnitude ϵ, we would sample to produce the set $y \in \{k\epsilon | k \in [-\lceil \frac{1}{\epsilon} \rceil, \lceil \frac{1}{\epsilon} \rceil] \cap \mathbb{Z}\}$, and for any $w \in y + [-\epsilon, \epsilon)$,we upper bound the instability at w with size of the image of $y + [-2\epsilon, 2\epsilon]$ under π [4].

[1] via Google Doc link, [redacted] shared with me on day23 month4 year2019.

[2] Later in this document, I switch from refering to "I" and "me" to "we" per typical academic stylings; the "we" that will be used refers only to me, the author.

[3] or, rather, an upper bound on the *in*-stability

[4] Why $[-2\epsilon, 2\epsilon]$? Because if $w \in y + [-\epsilon, \epsilon)$, then $w + [-\epsilon, \epsilon] \subseteq y + [-2\epsilon, 2\epsilon]$, and thus $\pi(w + [-\epsilon, \epsilon]) \subseteq \pi(y + [-2\epsilon, 2\epsilon])$.

3 Technical Details

3.1 The Policy Network, the Input / Observation Space, and the Output/Action Space

At the time of writting this document, we only discuss the policy network for the system driving straight. We are currently working to produce similar results for the circle-following network. These networks and the nature of their pre-processing and post-processing were devised by Edward Ahn to the best of my current knowledge.

For either network, the architecture is the same. The network consists of three layers, each fully connected; there are two 32-unit hidden layers with tanh activation functions, and a 2-unit linear output layer. The input space is five dimensional , representing the *relative* value of the vector (y, yaw, $\dot{x}$, $\dot{y}$, $y\dot{a}w$). This vector is relative to the line that is meant to be followed (which is, by default, is the line $y = 0$) - the relative nature of the observation is not critical for our purposes here, since the required transformation does not alter the possible input set to the policy, and thus does not effect our image calculations. For more details on the transformation of the input space when running in the environment, see [redacted]

The output space consists of the vector (vel, yaw). The raw output from the network is post-processed to be scaled correctly and in the acceptable range of yel $\in [0, 1.3]$ and yaw $\in [-\frac{\pi}{6}, \frac{\pi}{6}]$. The relavent scaling and clipping is implemented by **properlyTransformAction**, shown below. Notice, importantly for our future use, that **properlyTransformAction** is non-decreasing with thisAction, so the lexical ordering of the outputs is never the reverse of the lexical ordering of the inputs (i.e., if $\forall i \in [dim(\vec{x_1})].\{\vec{x_1}\}_i \leq \{\vec{x_2}\}_i$ then $\forall i \in [dim(\vec{x_1})].\{\textbf{properlyTransformAction}(\vec{x_1})\}_i \leq \{\textbf{properlyTransformAction}(\vec{x_2})\}_i$).

```
def properlyTransformAction(thisAction):
    lb = np.array([ 0.0 , -0.52359878]);
    ub = np.array([1.3, 0.52359878]);
    scaled_action = lb + (thisAction + 1.) * 0.5 * (ub - lb);
    scaled_action = np.clip(scaled_action, lb, ub);
    return scaled_action;
```

3.2 Propagating Boxes Through the Policy Network

NOTE: the process described here leverages the fact that we are dealing with a pre-trained, fixed-weight feed-forward nueral net. This would not, for instance, extend trivially to recurrent neural nets.

Under sensible choices for **measure**, then, this would imply that $\widetilde{\pi}(w) = \textbf{measure}(\pi(w + [-\epsilon, \epsilon])) \leq \textbf{measure}(\pi(y + [-2\epsilon, 2\epsilon]))$

In order to propagate boxes through the network, we need to examine how to transform a box once is passes through a unit in the network. That is, if $u : \mathbb{R}^{I_u} \to \mathbb{R}^{O_u}$ is a unit of the network, and $\times_{i \in I_u}[a_i, b_i]$ is the input box, we need to calcuate $u(\times_{i\in[I_u]}[a_i, b_i])$.

In the network in question, there are two types of activation functions: linear and tanh. Let $w \in \mathbb{R}^{I_u}$ be the weights of the unit, $b \in \mathbb{R}$ be the bias, and $x \in \mathbb{R}^{I_u}$ be the input value. We have that:

$$u_{linear}(x) =< w, x > +b$$
$$u_{tanh}(x) = tanh(< w, x > +b) = tanh(u_{linear}(x))$$

where, here, $< ., . >$ is the L2-norm. Notice that since $tanh$ is a strictly increasing function, the maximum and minimum of $u_{tanh}(x)$ occurs for the same candidate x values as the respective max or min as $u_{linear}(x)$. To find the inputs that produce the extreme values of the activation function over the input space, it thus suffices to find the values in $\times_{i\in[I_u]}[a_i, b_i]$ that maximize or minimize $< w, x >$. Trivial algrbra show that:

$$\underset{x\in\times_{i\in[I_u]}[a_i,b_i]}{argmin} < x, w >=< b_i\mathbb{1}(\{w\}_i \leq 0) + a_i\mathbb{1}(\{w\}_i > 0)|i \in [I_u] >$$

$$\underset{x\in\times_{i\in[I_u]}[a_i,b_i]}{argmax} < x, w >=< a_i\mathbb{1}(\{w\}_i \leq 0) + b_i\mathbb{1}(\{w\}_i > 0)|i \in [I_u] >$$

where here, $< z_i|i >$ is sequence construction notation. With this, we can compute the images of the input space under the activation functions as follows: for $u \in \{u_{linear}, u_{tanh}\}$,

$$u(\times_{i\in I_u}[a_i, b_i]) = [u(\underset{x\in\times_{i\in[I_u]}[a_i,b_i]}{argmin} < x, w >), u(\underset{x\in\times_{i\in[I_u]}[a_i,b_i]}{argmax} < x, w >)]$$

Having established how a box should be propagated through a unit in the network, it is trivial to extend this to propogating a box through the entire network. If $u_{i,j}$ is the i^{th} unit on the j^{th} layer, and $B_{i,j}$ is the input box to unit $u_{i,j}$, then one simply feeds the output box from one layer into the next as one would for a normal feed-forward network:

$$u_{i,j+1}(B_{i,j+1}) = u_{i,j+1}(\times_{h\in[I_{u_{i,j+1}}]}u_{h,j}(B_{h,j}))$$

This completes discussion of how the input box can be propogated through the network. Yet to consider, however, is how to push the box through the function **properlyTransformAction**. As noted before, however, **properlyTransformAction** respects partial ordering of the input vectors, thus,

$$\forall w \in \textbf{properlyTransformAction}(\times_{i\in[I_u]}[a_i, b_i]).$$
$$\forall h \in \{1, 2\}.\{\textbf{properlyTransformAction}(< a_i|i \in [I_u] >)\}_h \leq \{w\}_h$$
$$\leq \{\textbf{properlyTransformAction}(< b_i|i \in [I_u] >)\}_h.$$

or, put another way, if

$$A = \textbf{properlyTransformAction}(< a_i | i \in [I_u] >)$$

and

$$B = \textbf{properlyTransformAction}(< b_i | i \in [I_u] >)$$

, then we have:

$$\textbf{properlyTransformAction}(\times_{i \in [I_u]} [a_i, b_i]) \subseteq [\{A\}_1, \{B\}_1] \times [\{A\}_2, \{B\}_2]$$

Even more strongly, it can be seen that the following holds:

$$\textbf{properlyTransformAction}(\times_{i \in [I_u]} [a_i, b_i]) = [\{A\}_1, \{B\}_1] \times [\{A\}_2, \{B\}_2]$$

since $< a_i | i \in [I_u] > \in \times_{i \in [I_u]} [a_i, b_i]$ and $< b_i | i \in [I_u] > \in \times_{i \in [I_u]} [a_i, b_i]$.

3.3 Sampling the Input Space and Obtaining Upper-bounds on the Instability of the Input Space

In subsection section 3.2, it was established how a box may be pushed through our policy function, π. This provides stability information for a few point of the input space of π; we would like to garner such information over the entire input space $\mathscr{I}$ under the assumption that the set is bounded. To perform such an analysis, we take strategically selected samples from $\mathscr{I}$, partitioning $\mathscr{I}$ into sets located around our samples, then use the machinary developed in section 3.2 to examine the stability in each partition.

First, we choose what our $\mathscr{I}$ is. From page 2 of the document provided by Edward Ahn, we take the following to be a reasonable superset for $\mathscr{I}$ to exist in:

- $y \in [-0.25, 0.25]$ m
- $yaw \in [-\frac{\pi}{3}, \frac{\pi}{3}]$ rad
- $\dot{x} \in [0, 1.3]$ m/s
- $\dot{y} \in [-0.6, 0.6]$ m/s
- $y\dot{a}w \in [-2.0, 2.0]$ rad/s^2

In order to generate an easily understandable plots in these early investigations, we choose $\mathscr{I}$ to be

$$\mathscr{I} = \{w \in \mathbb{R}^5 | \{w\}_1 \in [-0.25, 0.25], \{w\}_2 \in [-pi/3, pi/3], \{w\}_3 = \dot{x}_H, \{w\}_4 = \dot{y}_H, \{w\}_5 = y\dot{a}w_H\}$$

where $\dot{x}_H$, $\dot{y}_H$, and $y\dot{a}w_H$ are hyperparameters which we vary between different runs in order to produce different plots (see the results section for more details on how we handle these hyperparameters). We chose to vary the "position-related" variables (y and yaw) since they are easier to interprete in terms of this experiment, in contrast to the *rate of change* of these quantities, as represented by the other three variables.

From here, we draw samples from $\mathscr{I}$ by drawing from a grid drawn over it. There is a trade-off between reducing the magnitude of perturbation we consider and the number of samples we need; the smaller the perturbations we consider, the more samples needed and vice-versa. Being that we consider a grid over $\mathscr{I}$, and $\mathscr{I}$ is a box, we can describe how we select coordinates of our samples independantly of one another. Define the following functions:

$$N(maxVal, minVal, \epsilon) \doteq \lceil \frac{maxVal - minVal}{2\epsilon} \rceil$$

$$c_k(maxVal, minVal, \epsilon) \doteq \frac{maxVal + minVal}{2} + 2\epsilon(\frac{N(maxVal, minVal, \epsilon)}{2} - k - \frac{1}{2})$$

$$\mathbf{sampleCoordinates}(maxVal, minVal, \epsilon) \doteq \cup_{k \in [N(maxVal, minVal, \epsilon)]}\{c_k(maxVal, minVal, \epsilon)\}$$

In words, $\mathbf{sampleCoordinates}(maxVal, minVal, \epsilon)$ produces a set of points, centered at the mean of maxVal and minVal, such that $[minVal, maxVal] \subseteq [c_{N(maxVal,minVal,\epsilon)} - \epsilon, c_0 + \epsilon]$ and $c_i - c_{i+1} = 2\epsilon$. We will return to the details of that in a moment. Our actual set of samples from $\mathscr{I}$ is as follows:

$$\mathbf{samples}(\mathscr{I}) =$$
$$\mathbf{sampleCoordinates}(-0.25, 0.25, \quad 0.015 * (0.25 - (-0.25)) \quad) \times$$
$$\mathbf{sampleCoordinates}(-\frac{\pi}{3}, \frac{\pi}{3}, \quad 0.015 * (\frac{\pi}{3} - (-\frac{\pi}{3})) \quad)$$

Notice that this means we consider ϵ to be roughly 1.5% of the range of the coordinates; for later reference, we will let ϵ_1 be this value $(0.015 * 2 * 0.25)$ on the first coordinate and ϵ_2 be this value on the second coordinate $(0.015 * 2 * \frac{\pi}{3})$.

We digress for a moment to briefly show that our choice of samples is appropraite and does as claimed. First, notice that

$$c_i > c_{i+1}$$

and

$$c_i - \epsilon = c_{i+1} + \epsilon$$

thus

$$[c_i - \epsilon, c_i + \epsilon] \cup [c_{i+1} - \epsilon, c_{i+1} + \epsilon] = [c_{i+1} - \epsilon, c_i + \epsilon]$$

which in turn, by trivial induction, means

$$\cup_{x \in \mathbf{sampleCoordinates}(maxVal,minVal,\epsilon)}[x - \epsilon, x + \epsilon] = [c_{N(maxVal,minVal,\epsilon)} - \epsilon, c_0 + \epsilon]$$

We see that this is a superset of $[minVal, maxVal]$ since basic algebra shows that $c_{N(maxVal,minVal,\epsilon)} - \epsilon \leq minVal$ and $maxVal \leq c_0 + \epsilon$. In addition to this, we see that

$$[c_i - \epsilon, c_i + \epsilon] \cap [c_{i+1} - \epsilon, c_{i+1} + \epsilon] = \{c_i - \epsilon\} = \{c_{i+1} + \epsilon\}$$

Thus, while this arrangement produces a covering of the set $[minVal, maxVal]$ when we center intervals of size 2ϵ at each sample, it also does not result in excessive overlap or excessive area outside of $[minVal, maxVal]$. This latter item - which translates in the set

$$(\cup_{x \in \mathbf{sampleCoordinates}(maxVal,minVal,\epsilon)}[x - \epsilon, x + \epsilon]) \setminus [minVal, maxVal]$$

being small - is a property that slightly more natural and naive methods of sampling lack .

Thus far, we have discussed which samples we collect. It remains to state what we do with those samples and how they produce the information we desired. Ultimately, we want to bound $\widetilde{\pi}(w)$ for $w \in \mathscr{I}$, where we take the measure of a box to be the sum of the range of its coordinates [5]. By construction, and as justified in the previous paragraph, there exists $c \in \mathbf{samples}(\mathscr{I})$ such that $w \in c + [-\epsilon_1, \epsilon_1] \times [-\epsilon_2, \epsilon_2]$. Thus, $w + [-\epsilon_1, \epsilon_1] \times [-\epsilon_2, \epsilon_2] \subset c + [-2\epsilon_1, 2\epsilon_1] \times [-2\epsilon_2, 2\epsilon_2]$, and in turn , this gives:

$$\widetilde{\pi}(w) \leq \mathbf{measure}(\pi(c + [-2\epsilon_1, 2\epsilon_1] \times [-2\epsilon_2, 2\epsilon_2]))$$

Thus, to bound the local instability across $\mathscr{I}$, for each $c \in \mathbf{sampleCoordinates}(maxVal, minVal, \epsilon)$ we bound the instability for each point in $c + [-\epsilon_1, \epsilon_1] \times [-\epsilon_2, \epsilon_2]$ by $\mathbf{measure}(\pi(c + [-2\epsilon_1, 2\epsilon_1] \times [-2\epsilon_2, 2\epsilon_2]))$, recalling that $\cup_{c \in \mathbf{sampleCoordinates}(maxVal, minVal, \epsilon)}(c + [-\epsilon_1, \epsilon_1] \times [-\epsilon_2, \epsilon_2]) \supseteq [minVal, maxVal]$. To get a grasp of these values, in the next section we produce heatmap of this bound on the local instabilty.

4 Results

As detailed in section section 3.3, we produce heatmaps of bounds on $\widetilde{\pi}(w)$ for each $w \in \mathscr{I}$. In order to make sense of the results more easily, we produce three heatmaps for each setting of the hyper-parameters $\dot{x}_H$, $\dot{y}_H$, and $y\dot{a}w_H$; if $[a_1, b_1] \times [a_2, b_2]$ is the box over the policy's output velocity and output steering angle after being fed a box centered around one of the samples we took, then we produce heatmaps for:

- Red: the normalized value of output velocity displacement: $\frac{b_1 - a_1}{\text{range of output velocity}} = \frac{b_1 - a_1}{0.5}$
- Blue: the normalized value of the output steering angle displacement $\frac{b_2 - a_2}{\text{range of output steering angle displacement}} = \frac{3(b_2 - a_2)}{2\pi}$
- Grey: the sum of the other two heatmaps, providing a weighted, L1 measure of displacement across the output box.

We include in this write-up several of the plots produced out of the hundreds available. We choose the hyper-parameters among the ranges listed in the bulleted list in section section 3.3. To expediate delivery of this document and to avoid purely implementation details, at the time being we spare discussion of how the hyperparameters were chosen for efficient exploration.

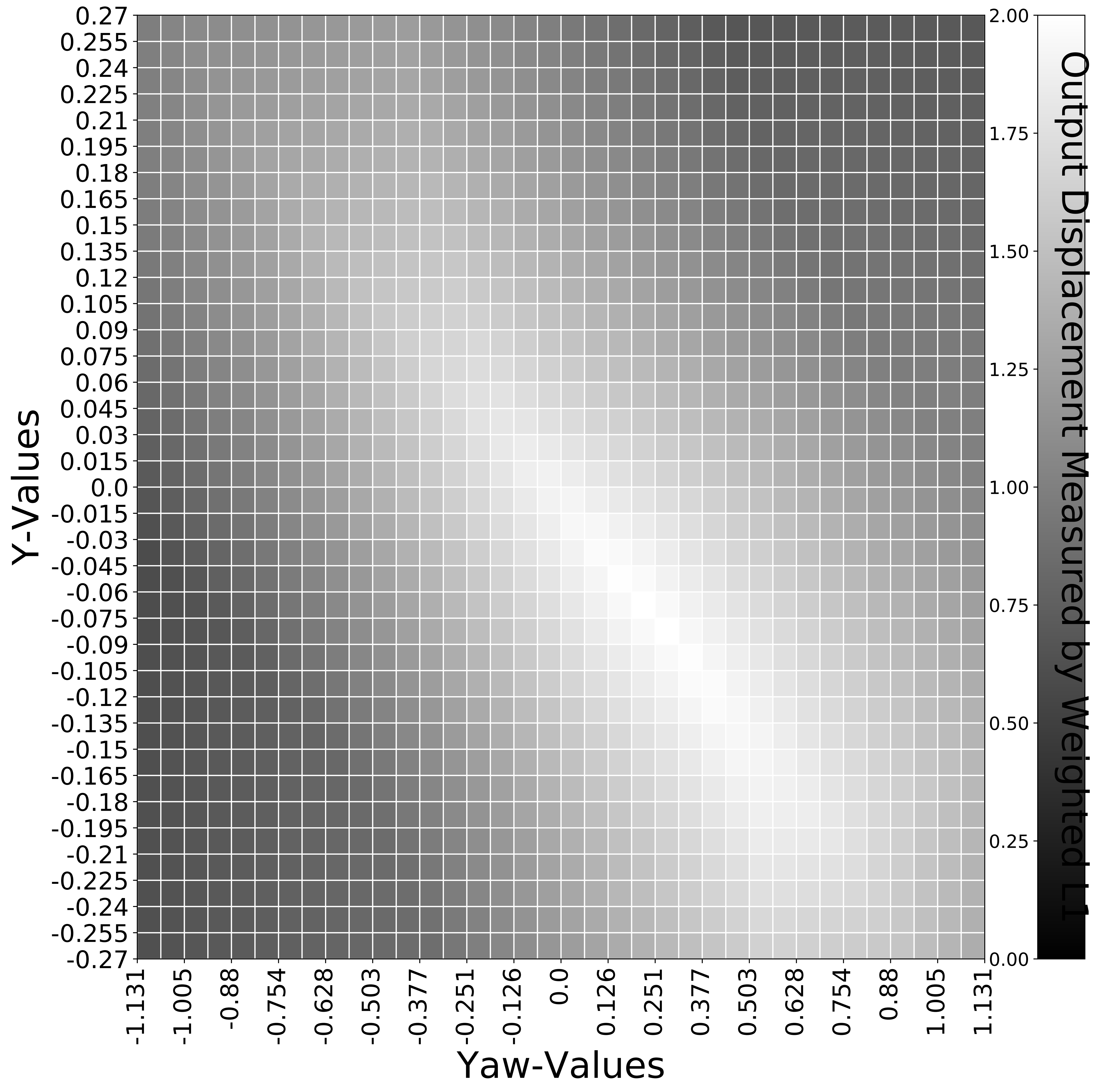
Among the results, notice that typically the pattern in the grey heatmap matches the pattern of the blue most closely. More encouragingly, notice that the regions of instability in the blue histogram is consistent with (but does not in itsself imply) the behavior of the agent trying to steer back towards the line (the line it is trying to follow is $y = 0$ - the input states are manipulated to maintain this illusions in all circumstances, as described in section 3.1). The red heatmap shows a variety of behavior.

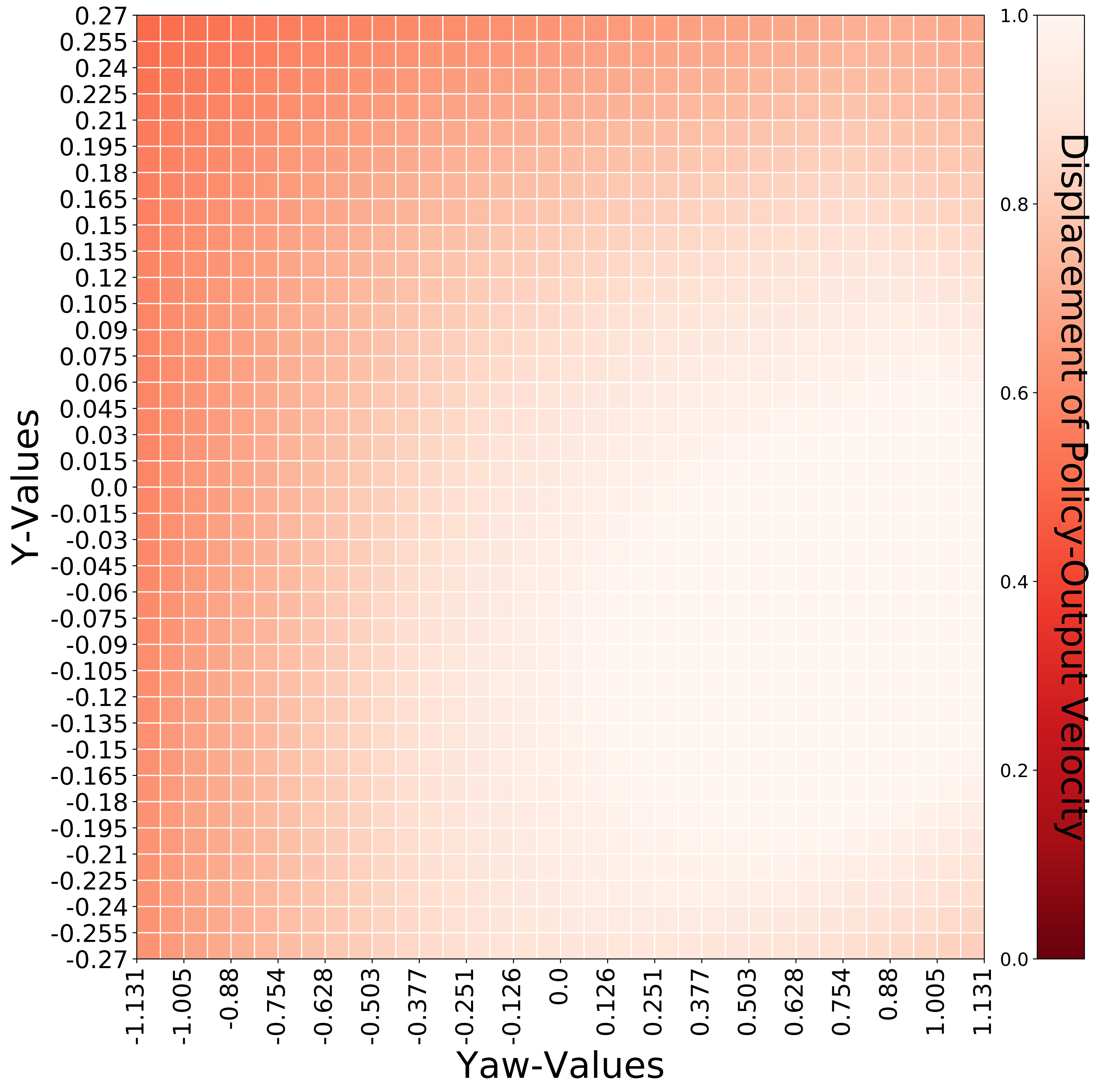
[5] i.e, $\mathbf{measure}(\times_{i \in [D]}[a_i, b_i]) = \sum_{i \in [D]}(b_i - a_i)$

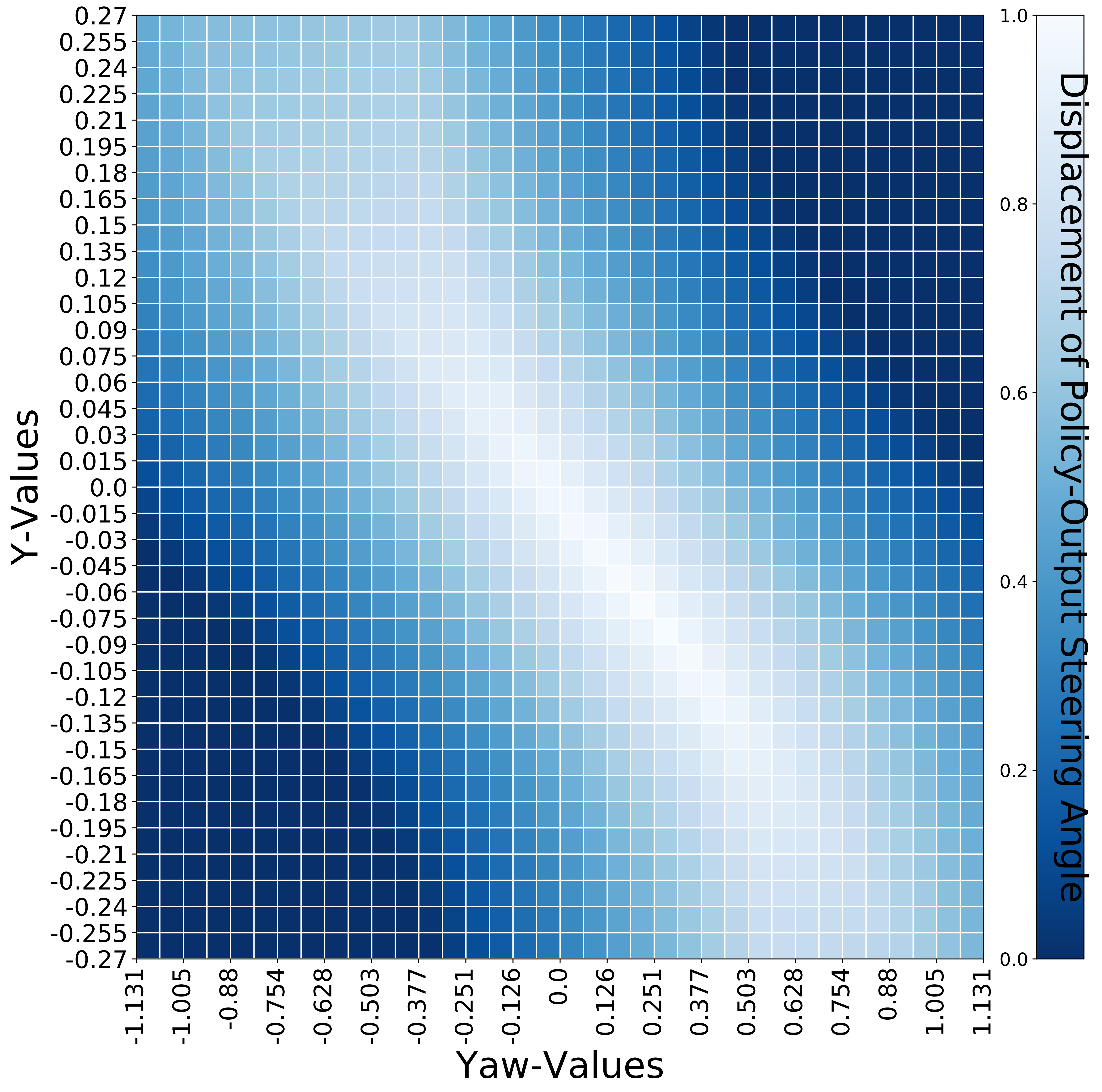
References

P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '77, pages 238–252, New York, NY, USA, 1977. ACM. doi: 10.1145/512950.512973. URL `http://doi.acm.org/10.1145/512950.512973`.

xDot:0, yDot:-0.07499, yawDot: -0.75, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
ee2424cc-9fed-410f-b586-886b60d43e5f



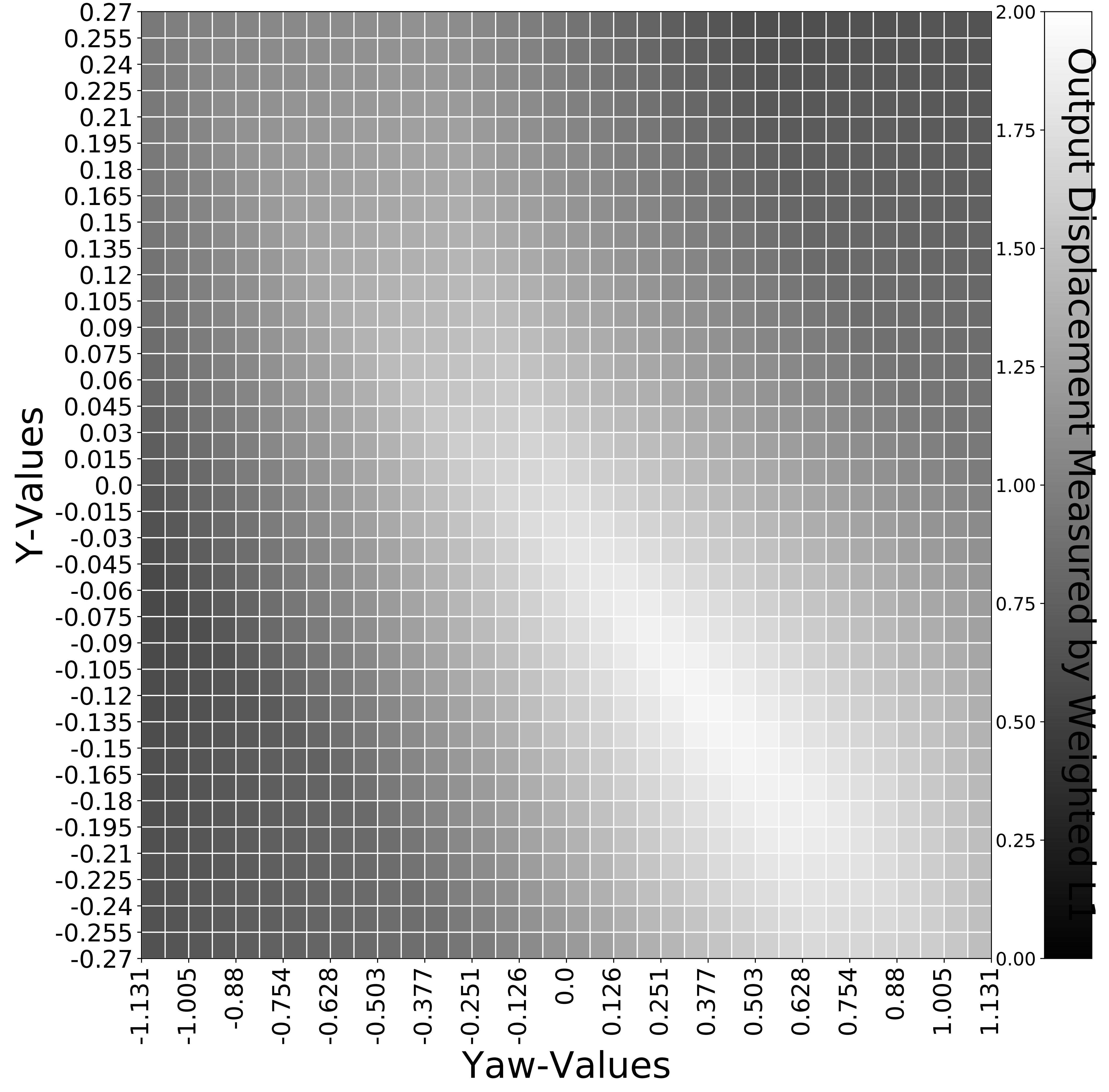


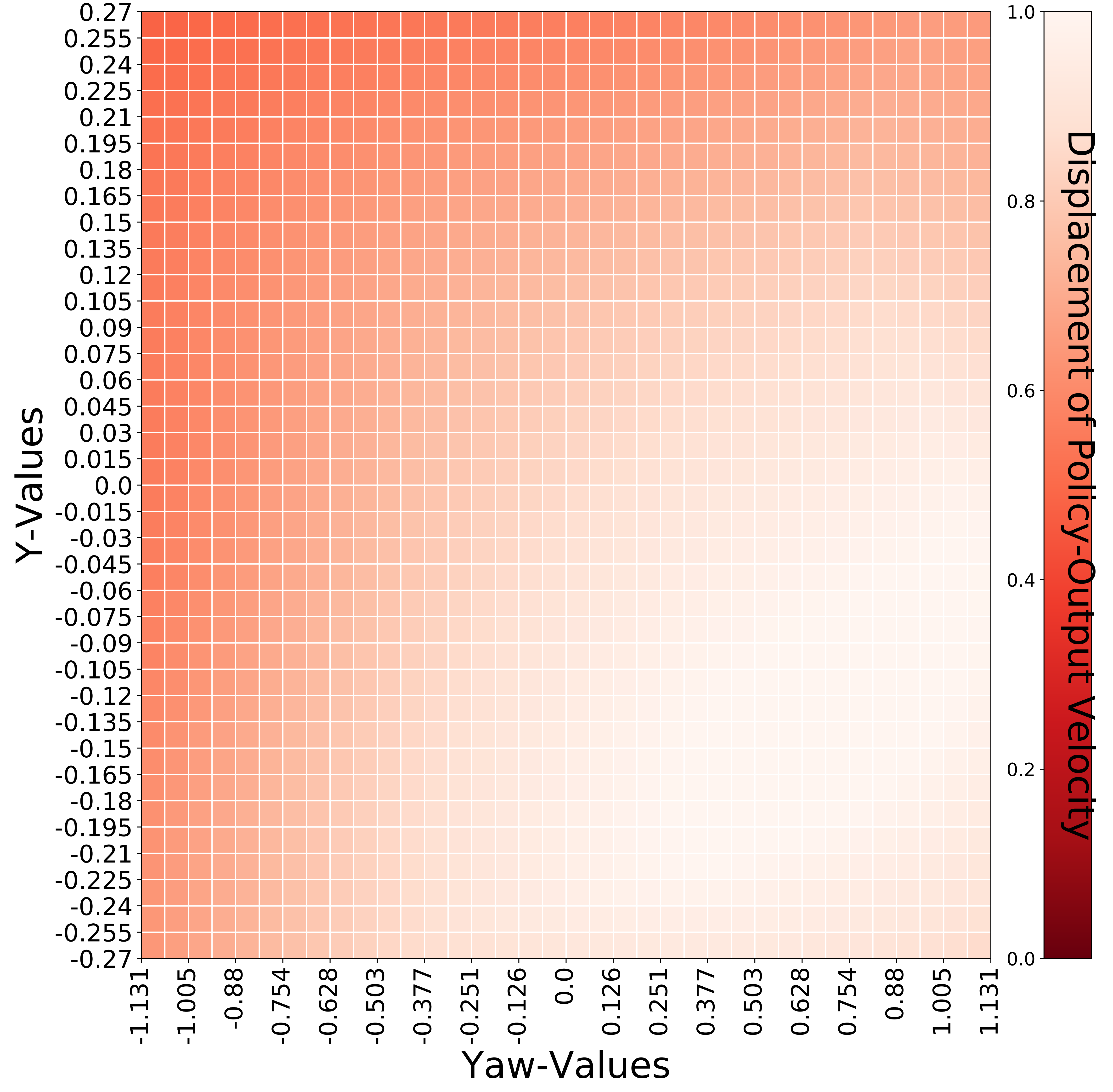


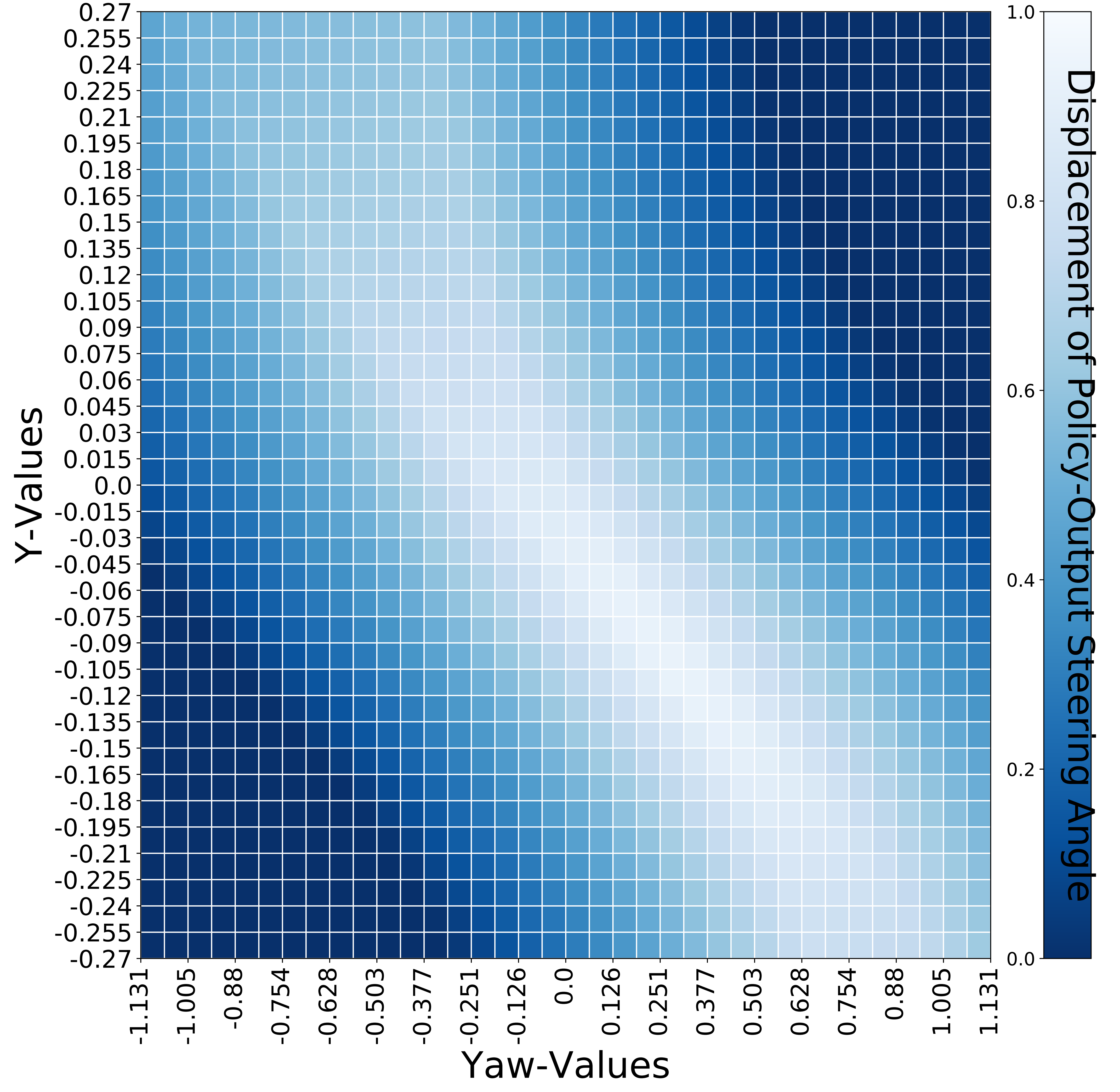
xDot:0, yDot:-0.375, yawDot: -0.5, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
e63fd2ec-4dcb-4e48-b311-b22abfa7d618

Y-Values
0.27
0.255
0.24
0.225
0.21
0.195
0.18
0.165
0.15
0.135
0.12
0.105
0.09
0.075
0.06
0.045
0.03
0.015
0.0
-0.015
-0.03
-0.045
-0.06
-0.075
-0.09
-0.105
-0.12
-0.135
-0.15
-0.165
-0.18
-0.195
-0.21
-0.225
-0.24
-0.255
-0.27
Yaw-Values
-1.131
-1.005
-0.88
-0.754
-0.628
-0.503
-0.377
-0.251
-0.126
0.0
0.126
0.251
0.377
0.503
0.628
0.754
0.88
1.005
1.131
Output Displacement Measured by Weighted L1
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
1.0
0.8
0.6
0.4
0.2
0.0

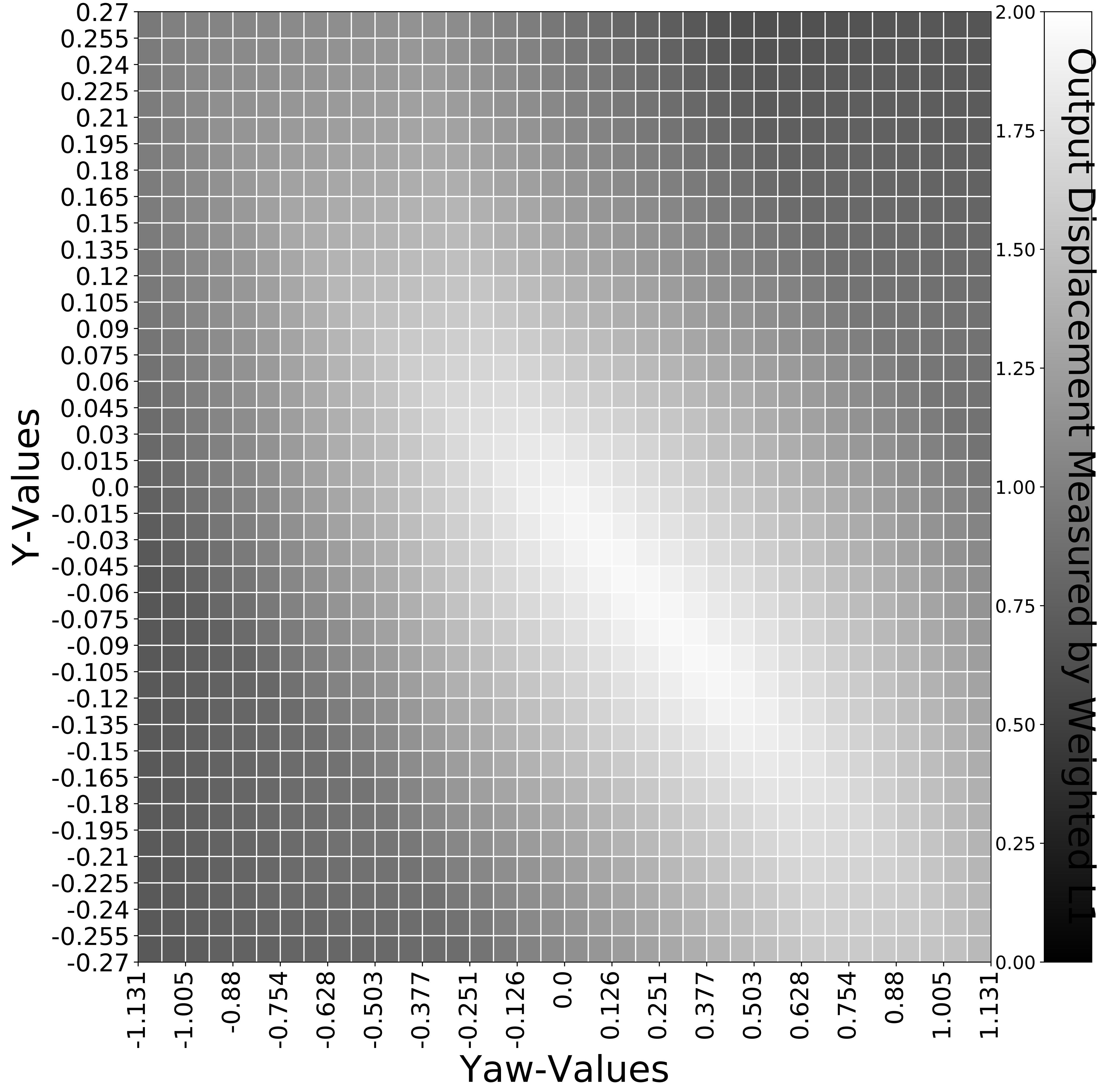
xDot:0, yDot:-0.22499, yawDot: -1.5, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
92fa1ea5-c02e-4d39-bbaa-2e45b6c84933

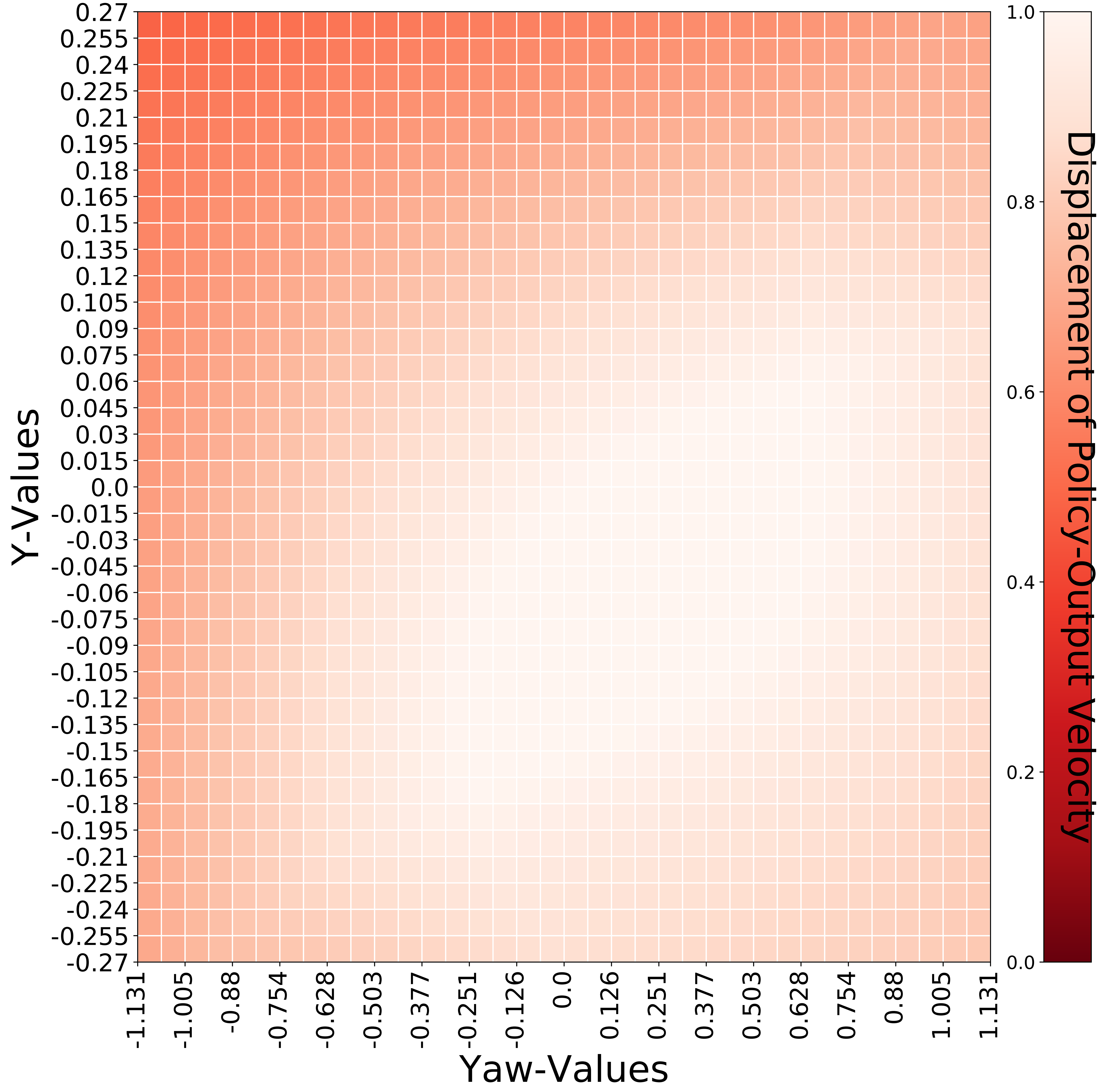


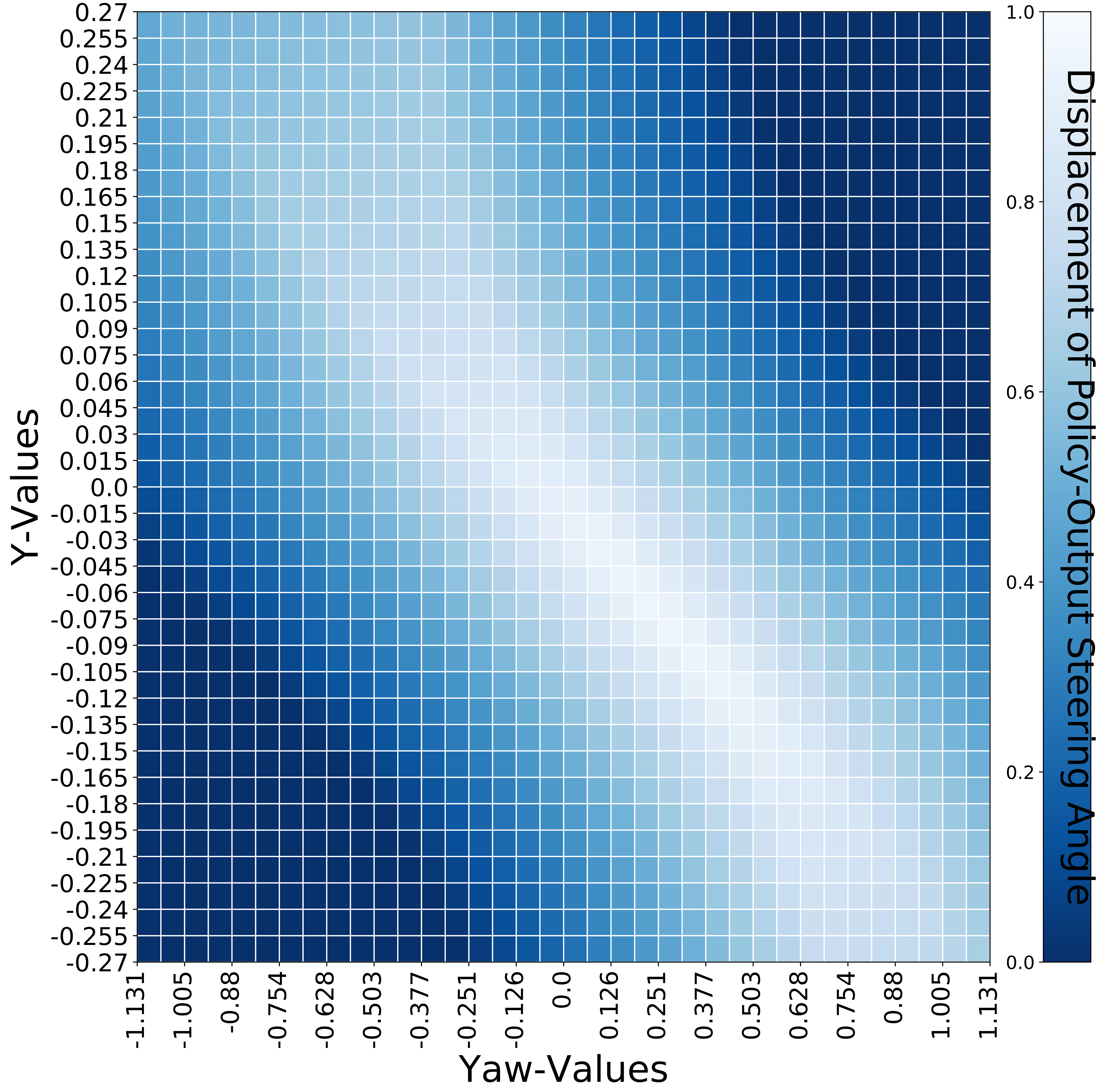




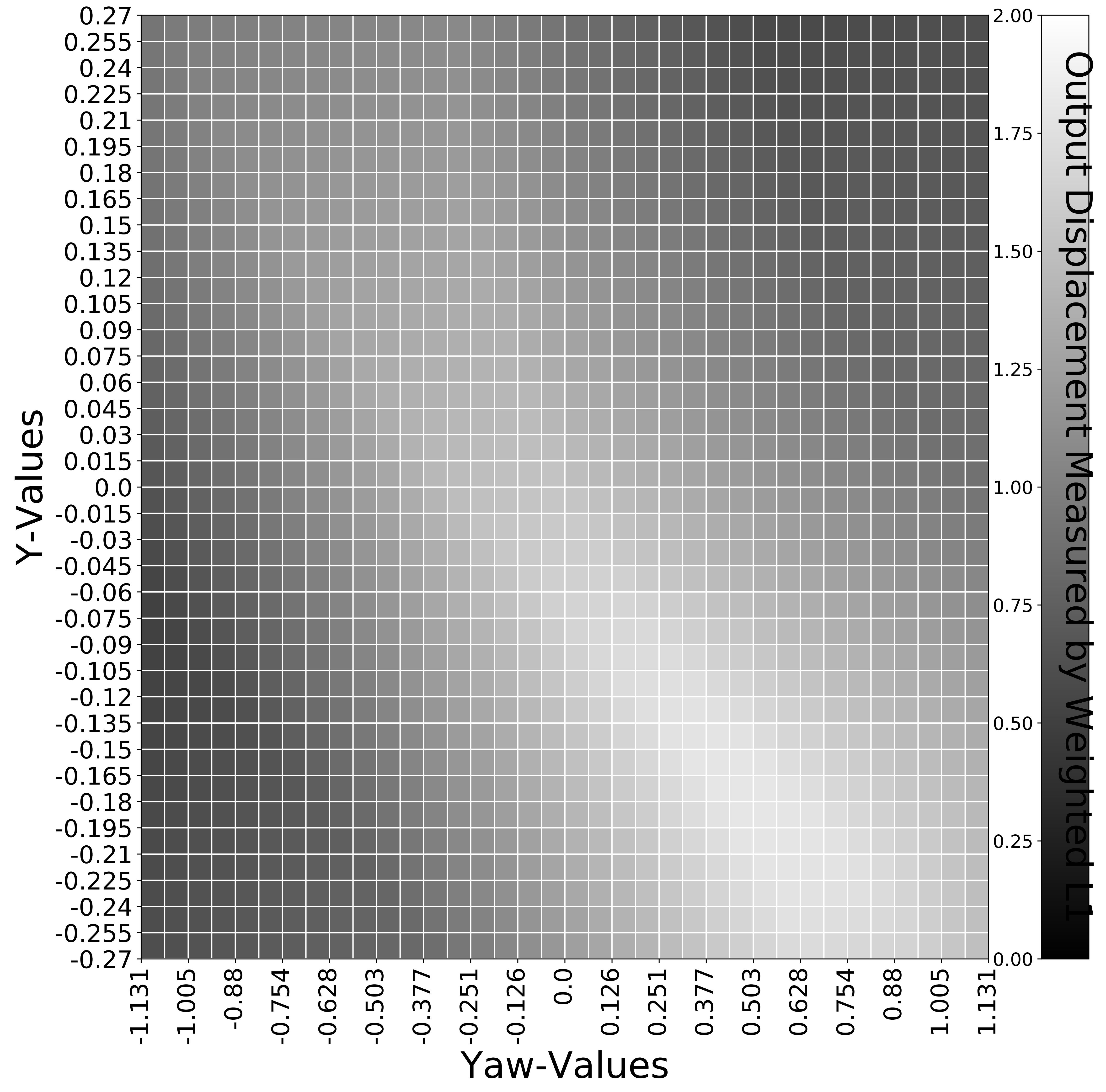
xDot:0, yDot:-0.525, yawDot: -1.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
12e7fb5f-1c28-40f8-bdc1-a393b2acccee

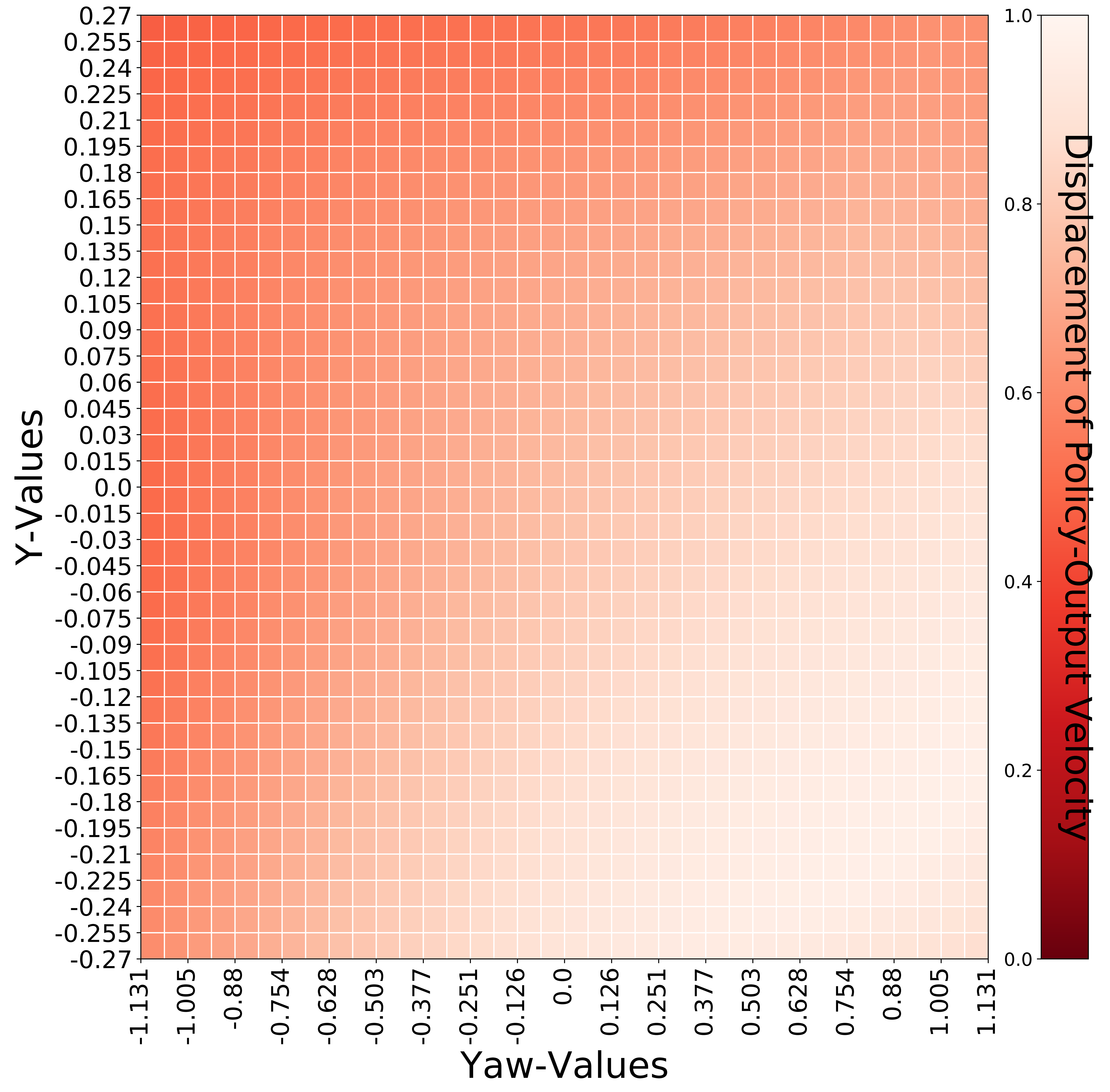


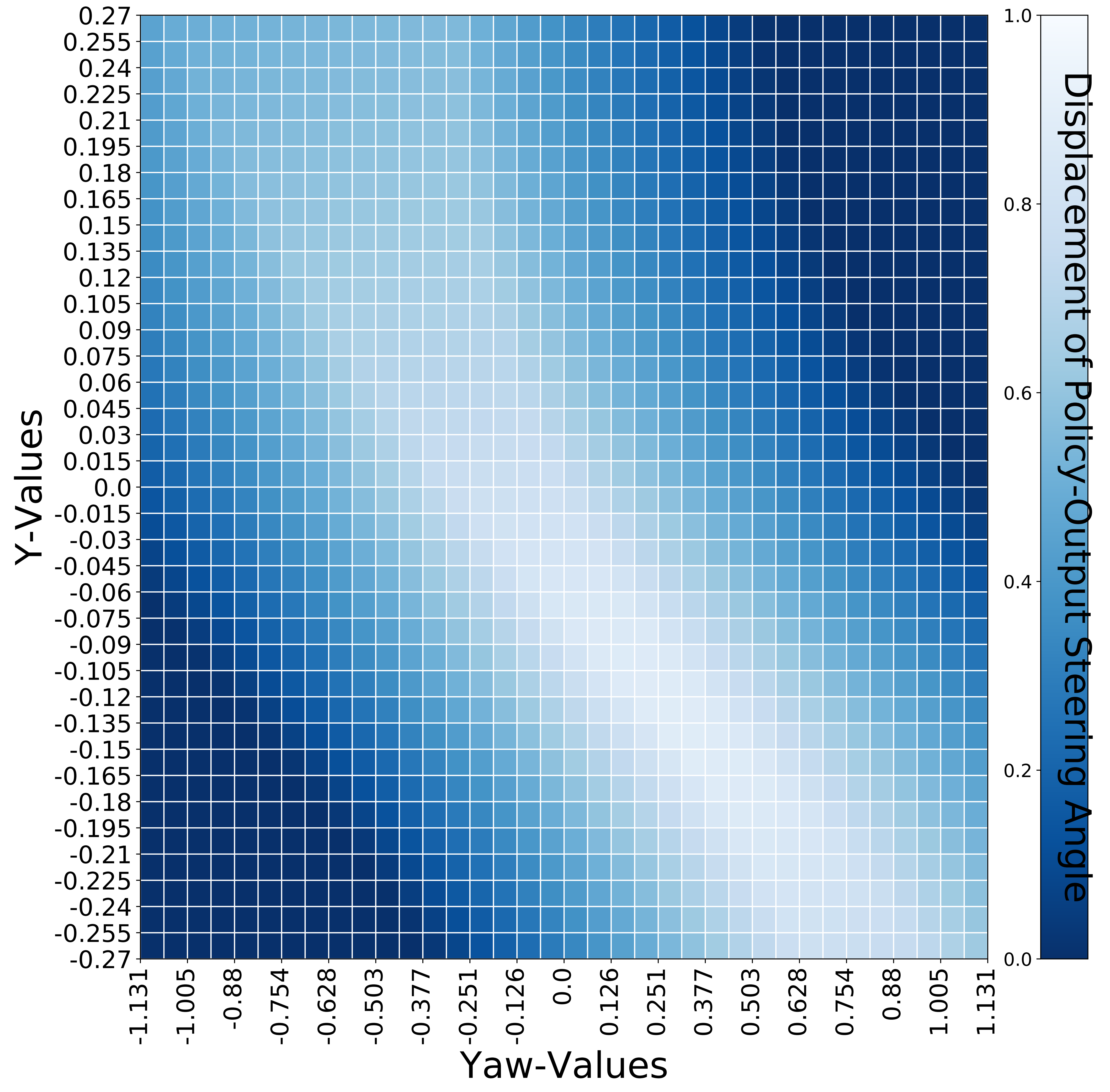




xDot:0, yDot:-0.15000, yawDot: -2.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
6400b846-944e-4192-bb52-16dc59b74545







xDot:0, yDot:-0.44999, yawDot: 2.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
0af9a686-8e25-45ce-90ab-97925cda6680

Y-Values
Yaw-Values
Output Displacement Measured by Weighted L1
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
0.27
0.0
-0.27
-1.131
0.0
1.131
0.00
2.00
0.0
1.0

xDot:0, yDot:-0.3, yawDot: -0.25, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
19c866e4-77f2-485d-8262-2e5bbfd9aed0

Y-Values
Yaw-Values
0.27
0.255
0.24
0.225
0.21
0.195
0.18
0.165
0.15
0.135
0.12
0.105
0.09
0.075
0.06
0.045
0.03
0.015
0.0
-0.015
-0.03
-0.045
-0.06
-0.075
-0.09
-0.105
-0.12
-0.135
-0.15
-0.165
-0.18
-0.195
-0.21
-0.225
-0.24
-0.255
-0.27
-1.131
-1.005
-0.88
-0.754
-0.628
-0.503
-0.377
-0.251
-0.126
0.0
0.126
0.251
0.377
0.503
0.628
0.754
0.88
1.005
1.131
Output Displacement Measured by Weighted L1
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
1.0
0.8
0.6
0.4
0.2
0.0

xDot:0, yDot:-0.6, yawDot: -1.25, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
0eb728ef-4e83-4ff0-825f-323694f3395f

Y-Values
Yaw-Values
Output Displacement Measured by Weighted L1
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
0.27
-0.27
-1.131
1.131
0.00
2.00
0.0
1.0

xDot:0, yDot:0.6, yawDot: -0.75, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
23d974c5-acbe-4006-841e-fdfe4be93408

Y-Values
Yaw-Values
Output Displacement Measured by Weighted L1
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
0.27
0.0
-0.27
-1.131
0.0
1.131
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
0.0
0.2
0.4
0.6
0.8
1.0

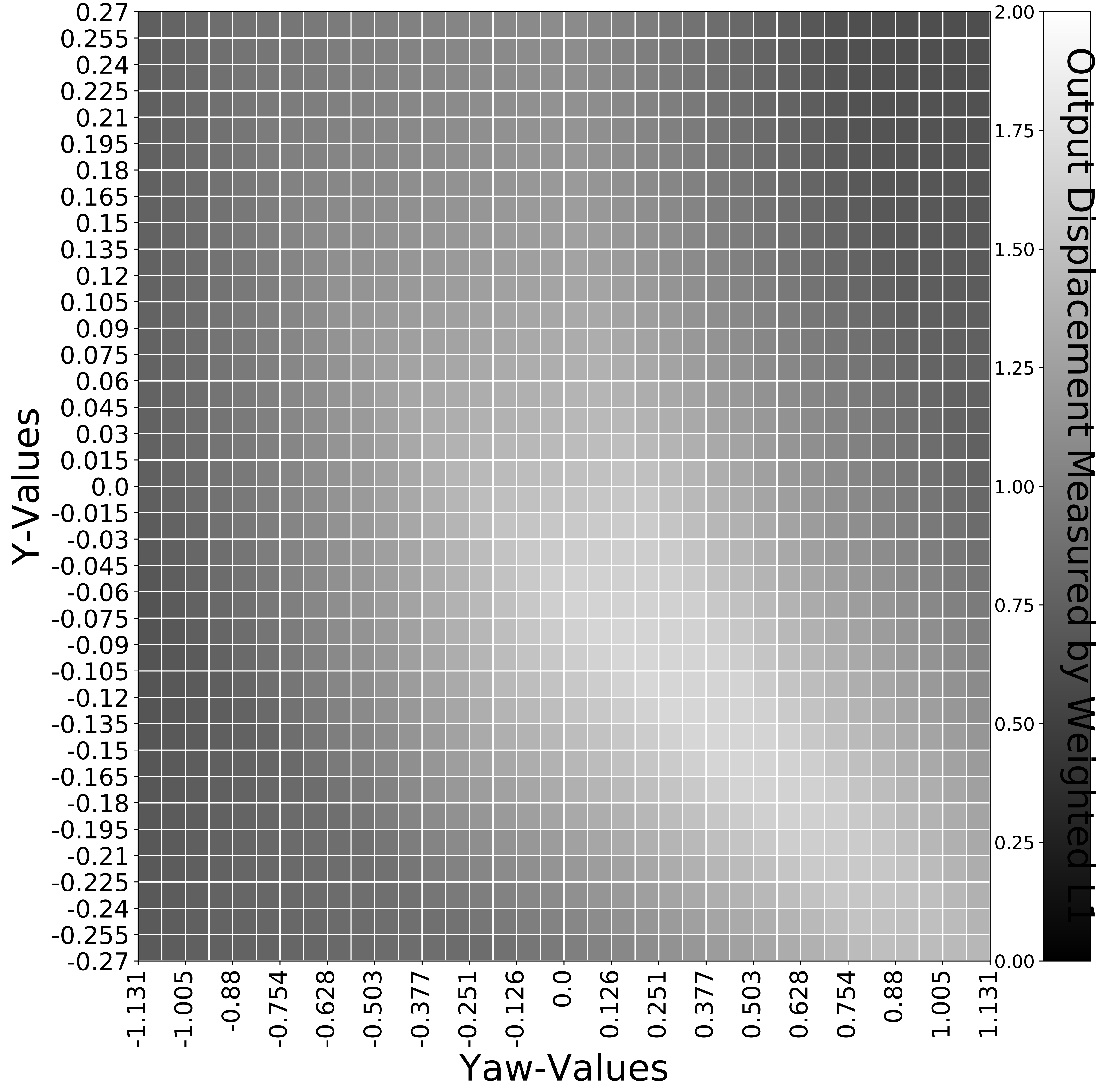
xDot:1.3, yDot:-0.07499, yawDot: -0.75, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
8af14347-1681-4d61-ab72-fa9d684bef17

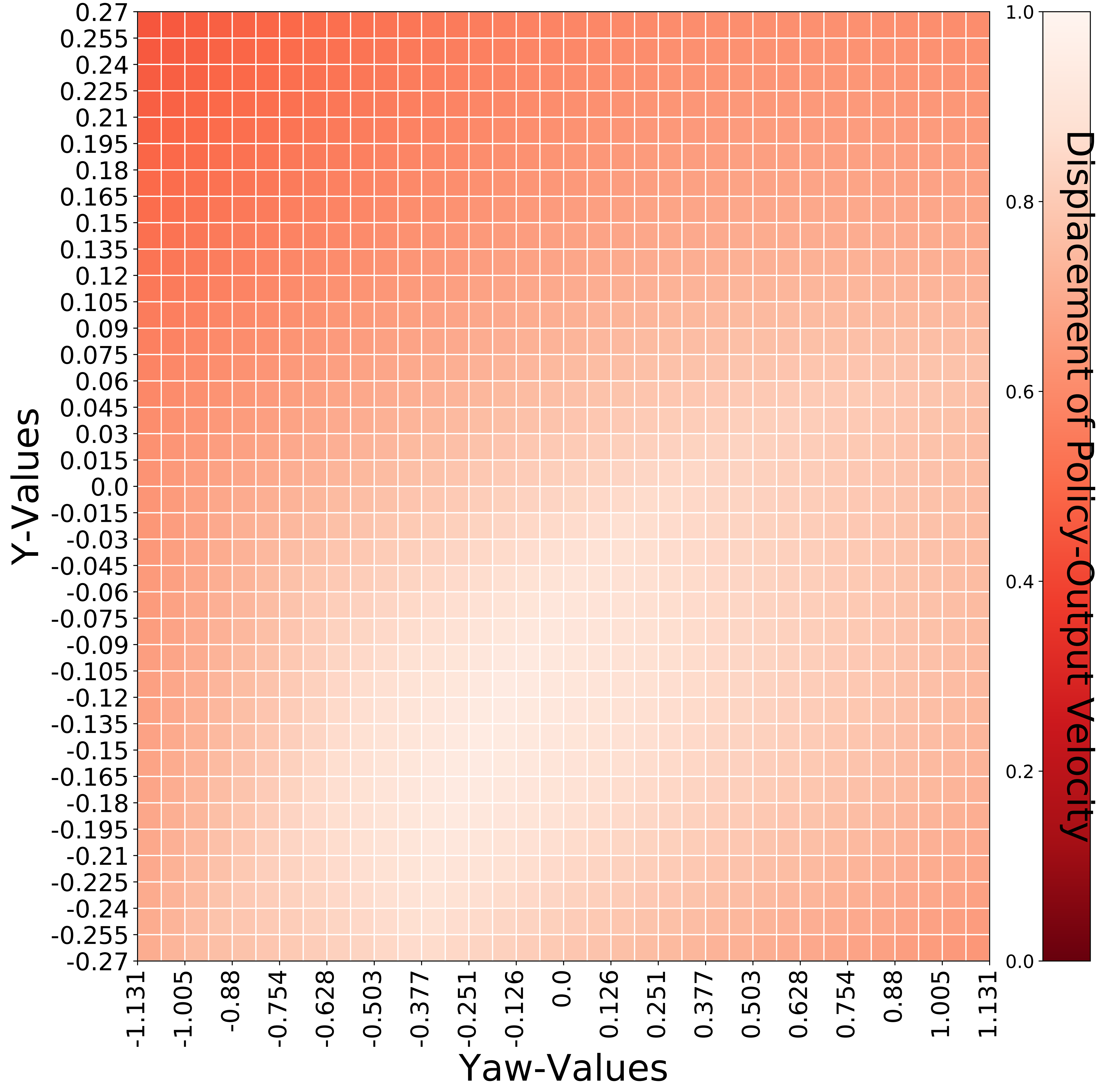
Y-Values
Yaw-Values
Output Displacement Measured by Weighted L1
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
0.27 0.255 0.24 0.225 0.21 0.195 0.18 0.165 0.15 0.135 0.12 0.105 0.09 0.075 0.06 0.045 0.03 0.015 0.0 -0.015 -0.03 -0.045 -0.06 -0.075 -0.09 -0.105 -0.12 -0.135 -0.15 -0.165 -0.18 -0.195 -0.21 -0.225 -0.24 -0.255 -0.27
-1.131 -1.005 -0.88 -0.754 -0.628 -0.503 -0.377 -0.251 -0.126 0.0 0.126 0.251 0.377 0.503 0.628 0.754 0.88 1.005 1.131
0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00
0.0 0.2 0.4 0.6 0.8 1.0

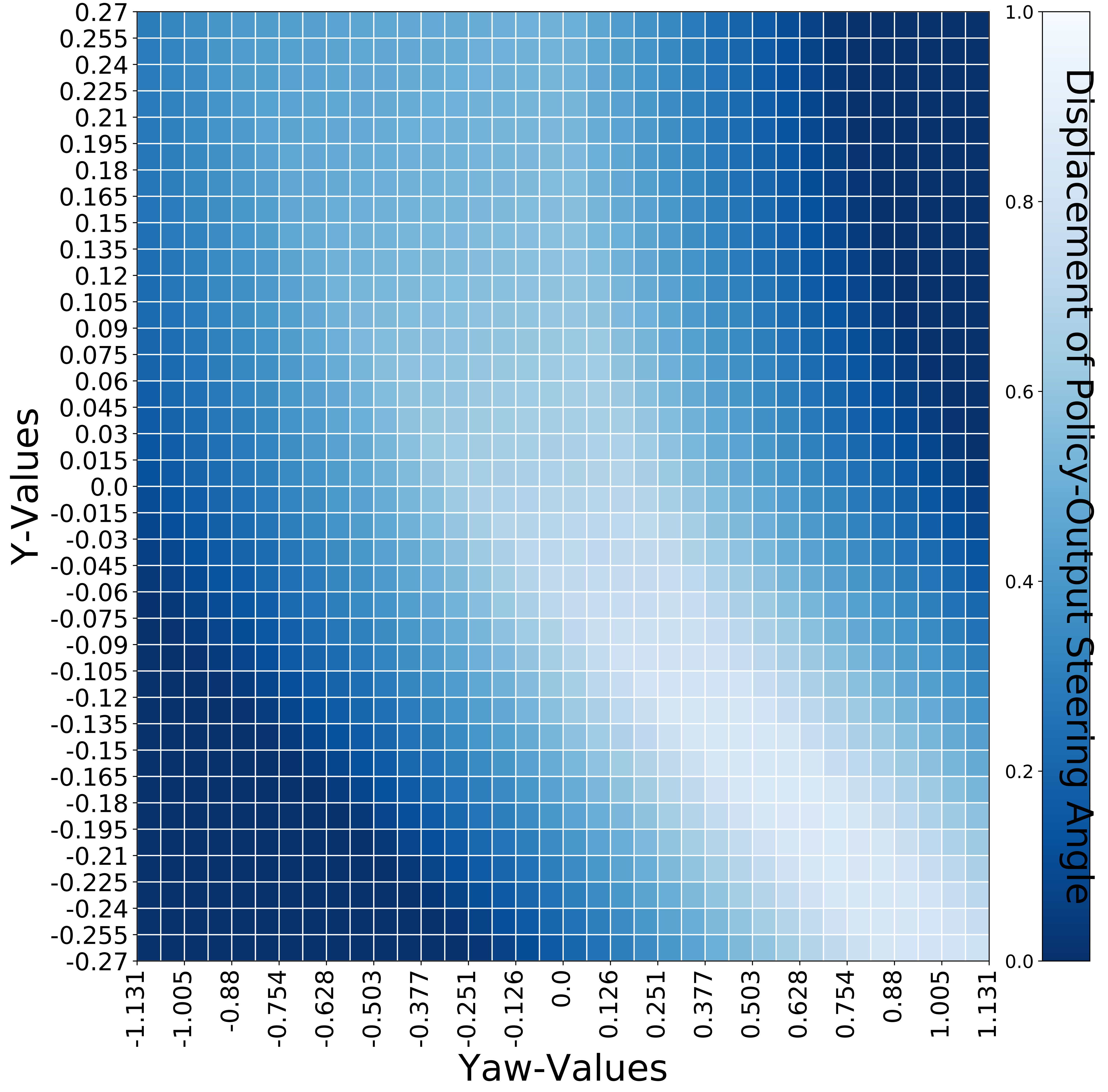
xDot:1.3, yDot:-0.22499, yawDot: -1.5, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
a73d7008-2de1-4b45-a198-2ae1fef1f31b

Y-Values
Yaw-Values
-1.131
-1.005
-0.88
-0.754
-0.628
-0.503
-0.377
-0.251
-0.126
0.0
0.126
0.251
0.377
0.503
0.628
0.754
0.88
1.005
1.131
0.27
0.255
0.24
0.225
0.21
0.195
0.18
0.165
0.15
0.135
0.12
0.105
0.09
0.075
0.06
0.045
0.03
0.015
0.0
-0.015
-0.03
-0.045
-0.06
-0.075
-0.09
-0.105
-0.12
-0.135
-0.15
-0.165
-0.18
-0.195
-0.21
-0.225
-0.24
-0.255
-0.27
Output Displacement Measured by Weighted L1
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Displacement of Policy-Output Velocity
Displacement of Policy-Output Steering Angle
0.0
0.2
0.4
0.6
0.8
1.0

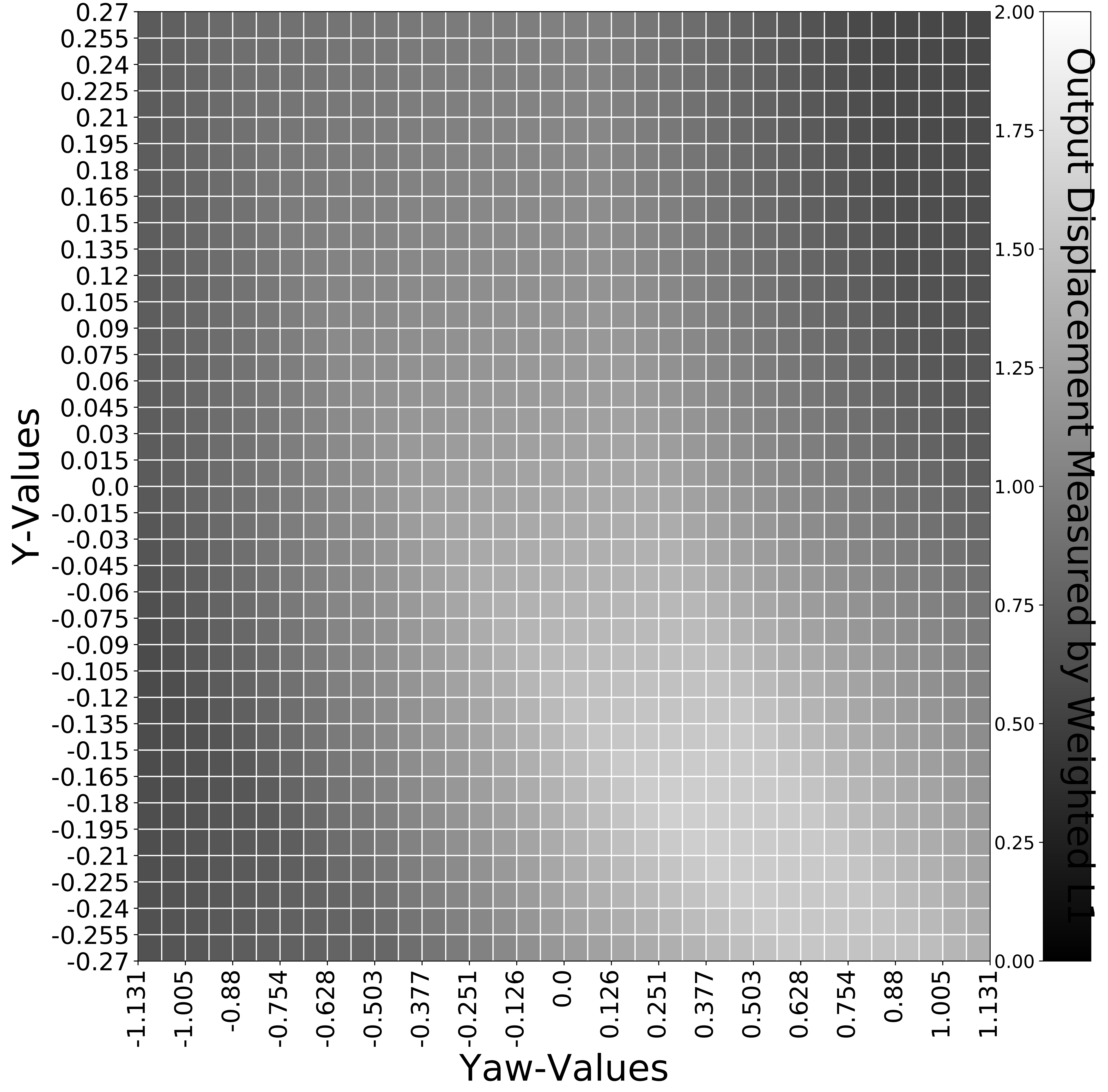
xDot:1.3, yDot:-0.525, yawDot: -1.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
954be364-0599-47e1-ac59-4b7b9c5ebd8b

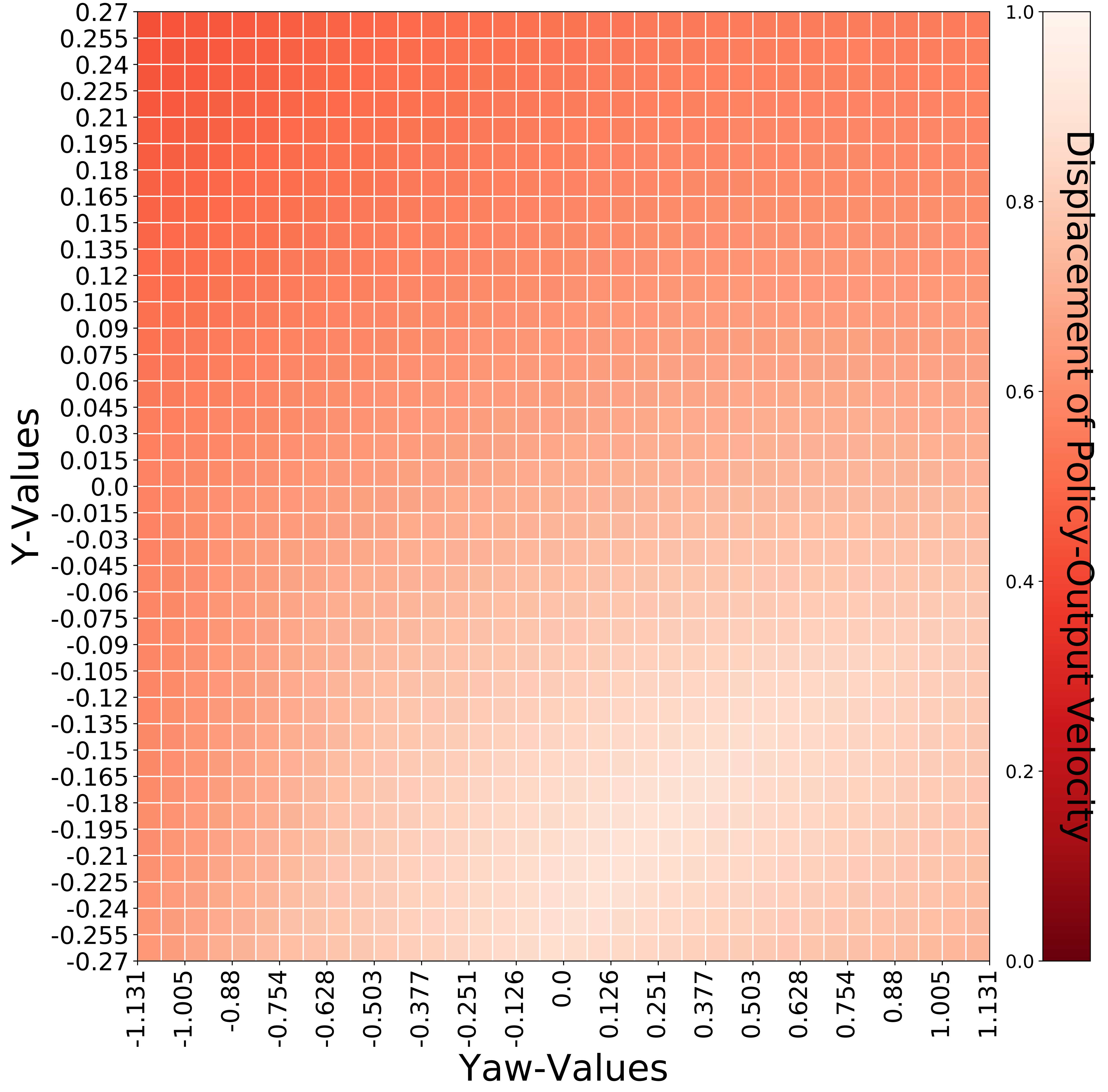


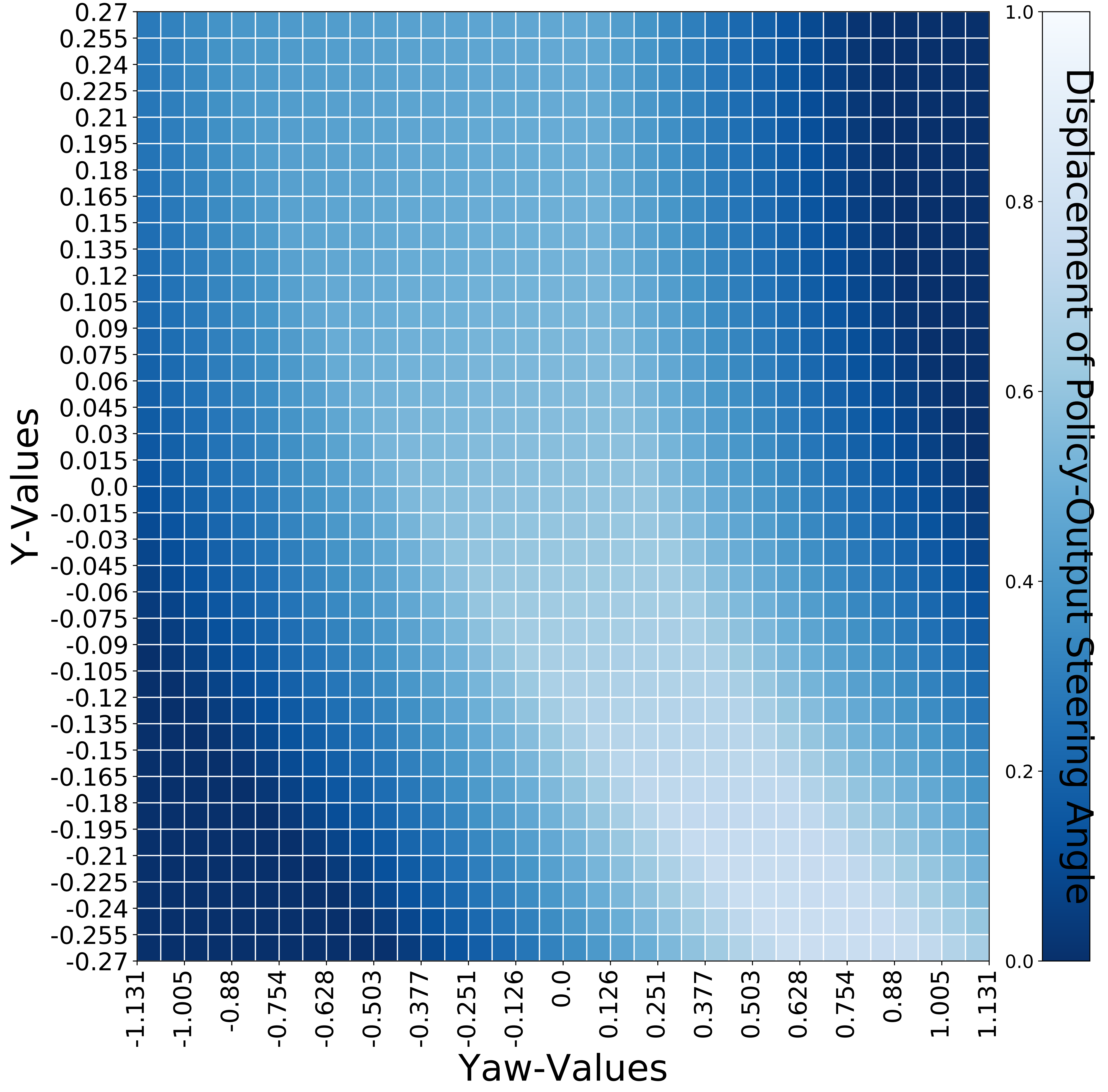




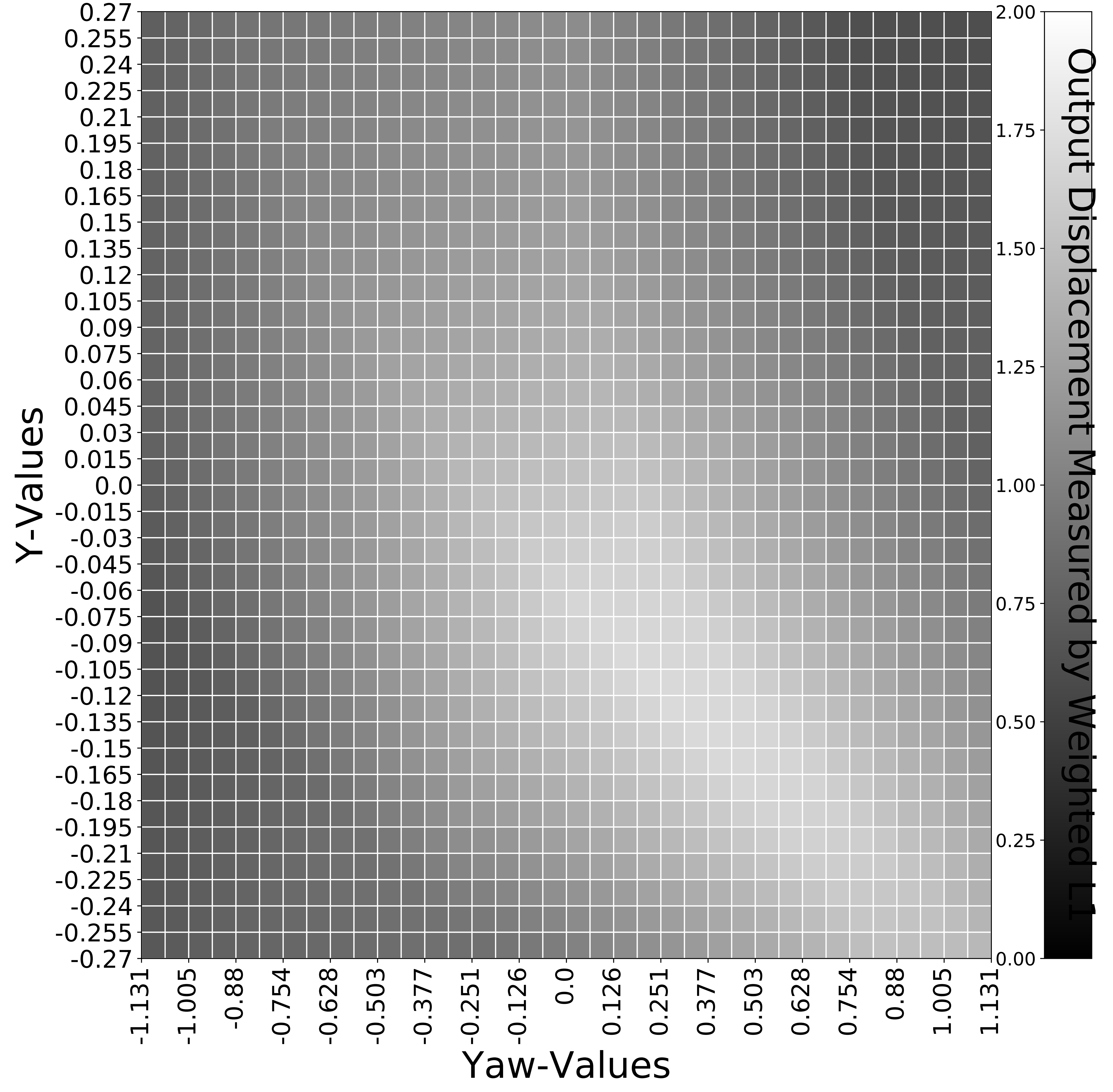
xDot:1.3, yDot:-0.525, yawDot: -2.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
d4d5c162-8fa2-4011-9ae2-7c60c5dce63b

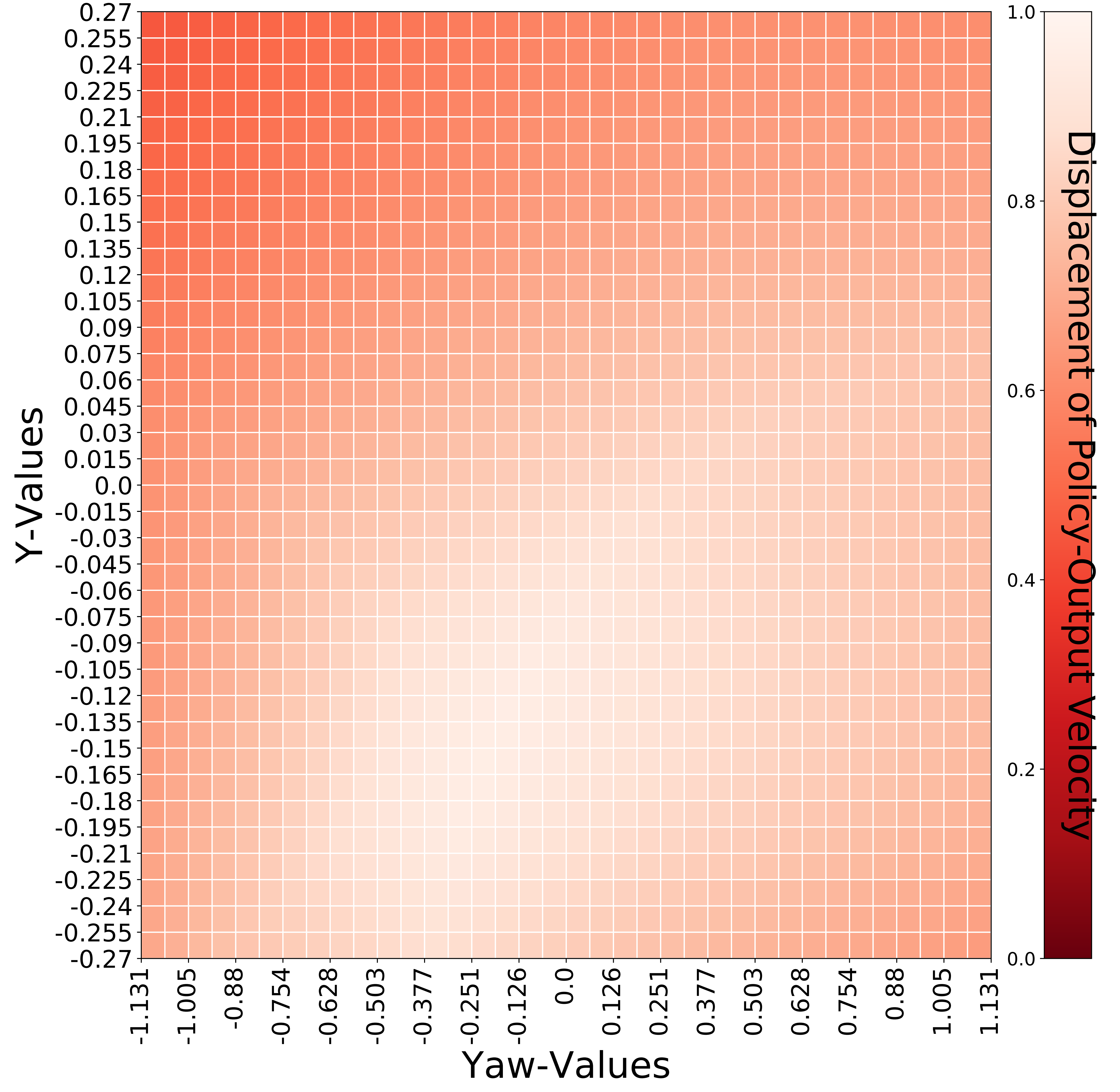


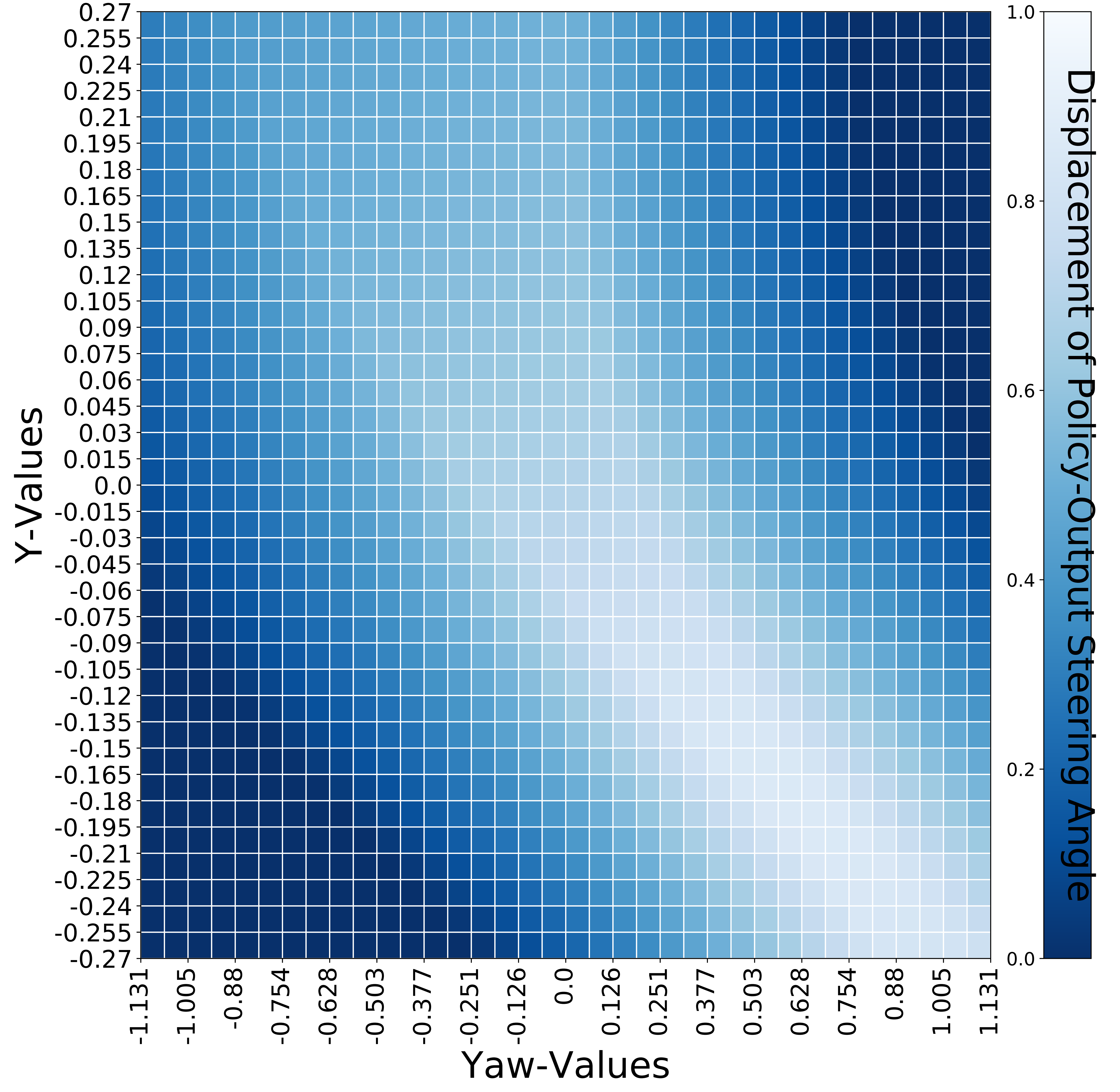




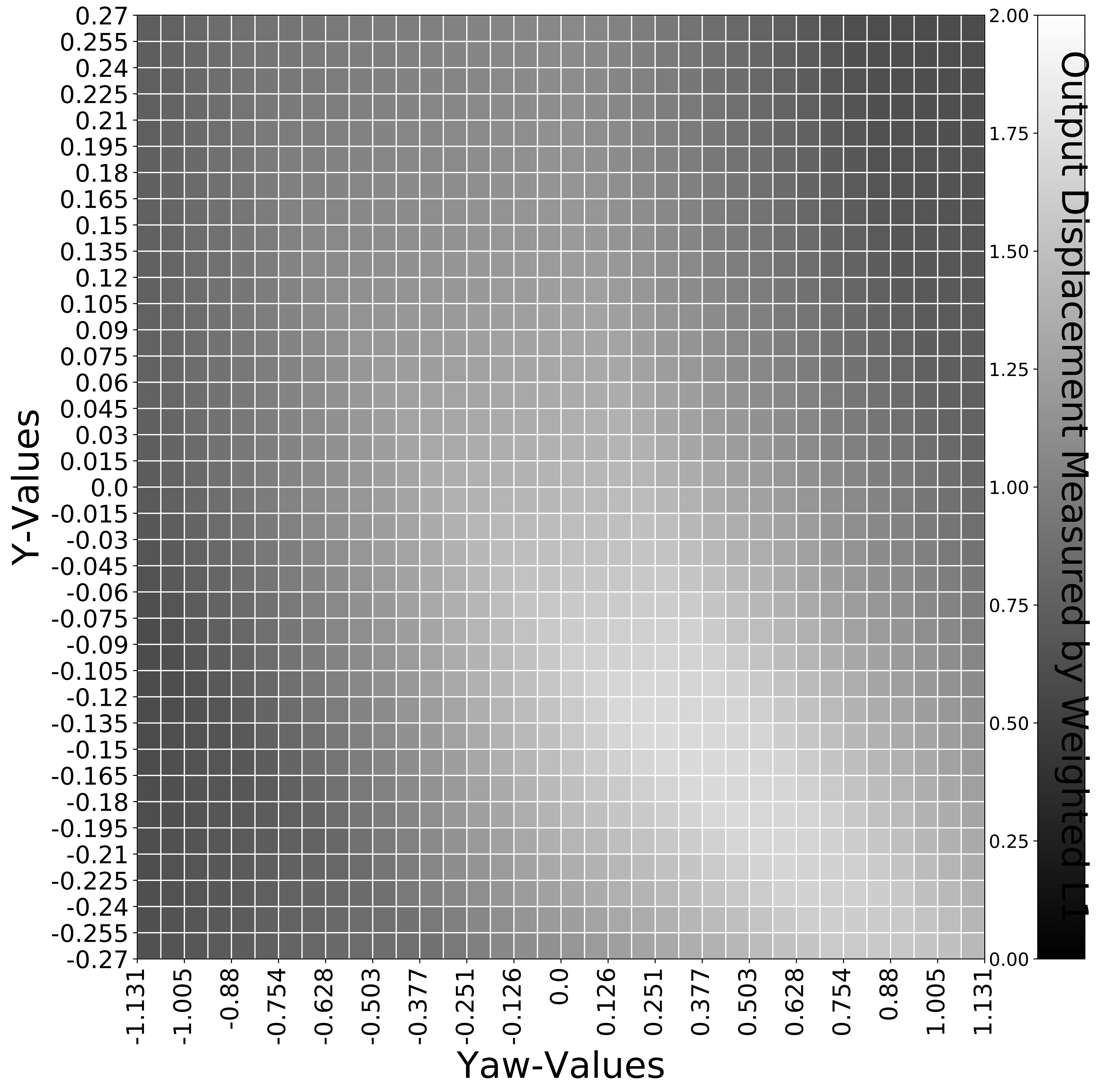
xDot:1.3, yDot:-0.44999, yawDot: -1.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
46393f28-4c07-4ac2-beb1-be0ae046d49d

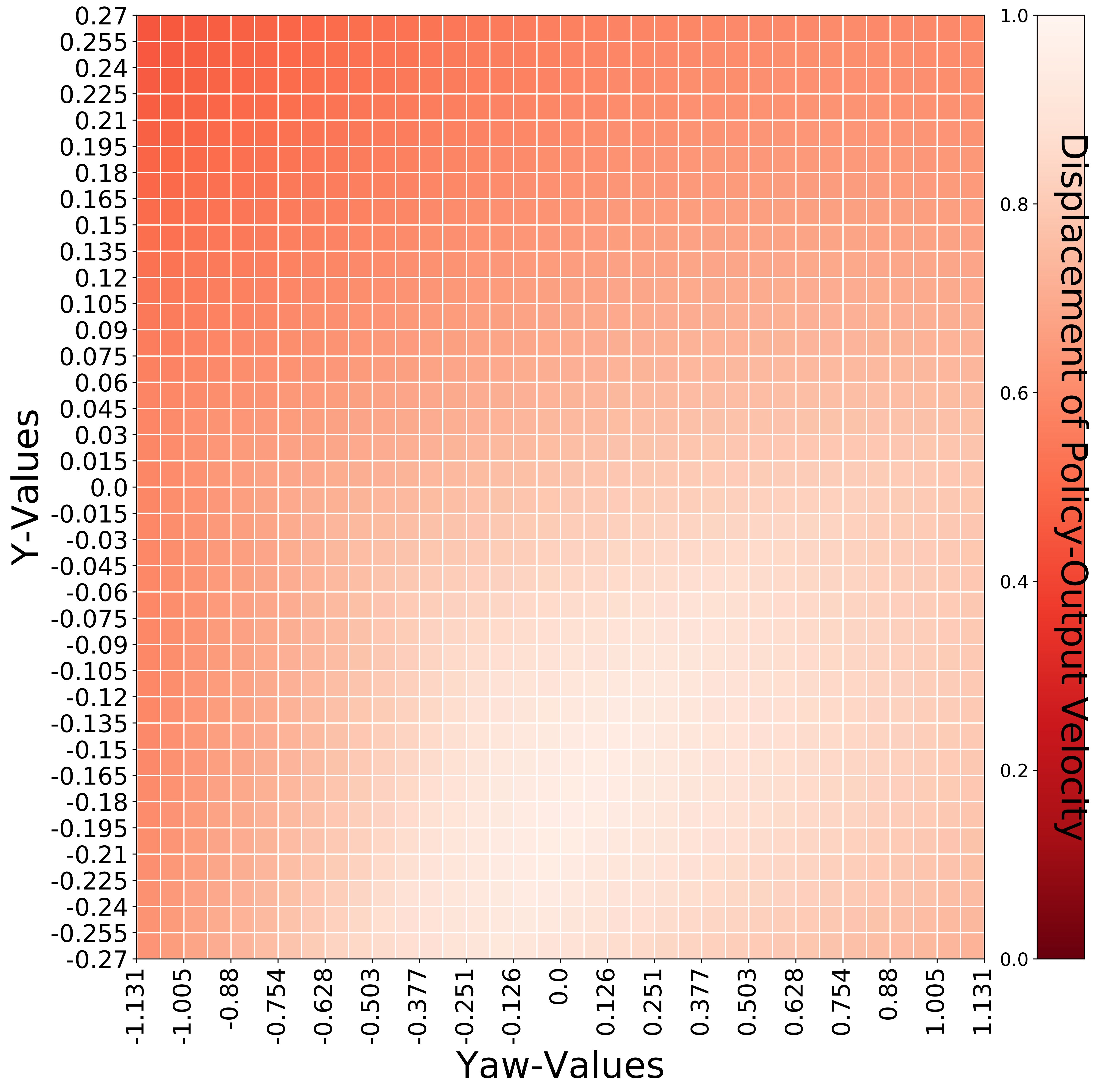


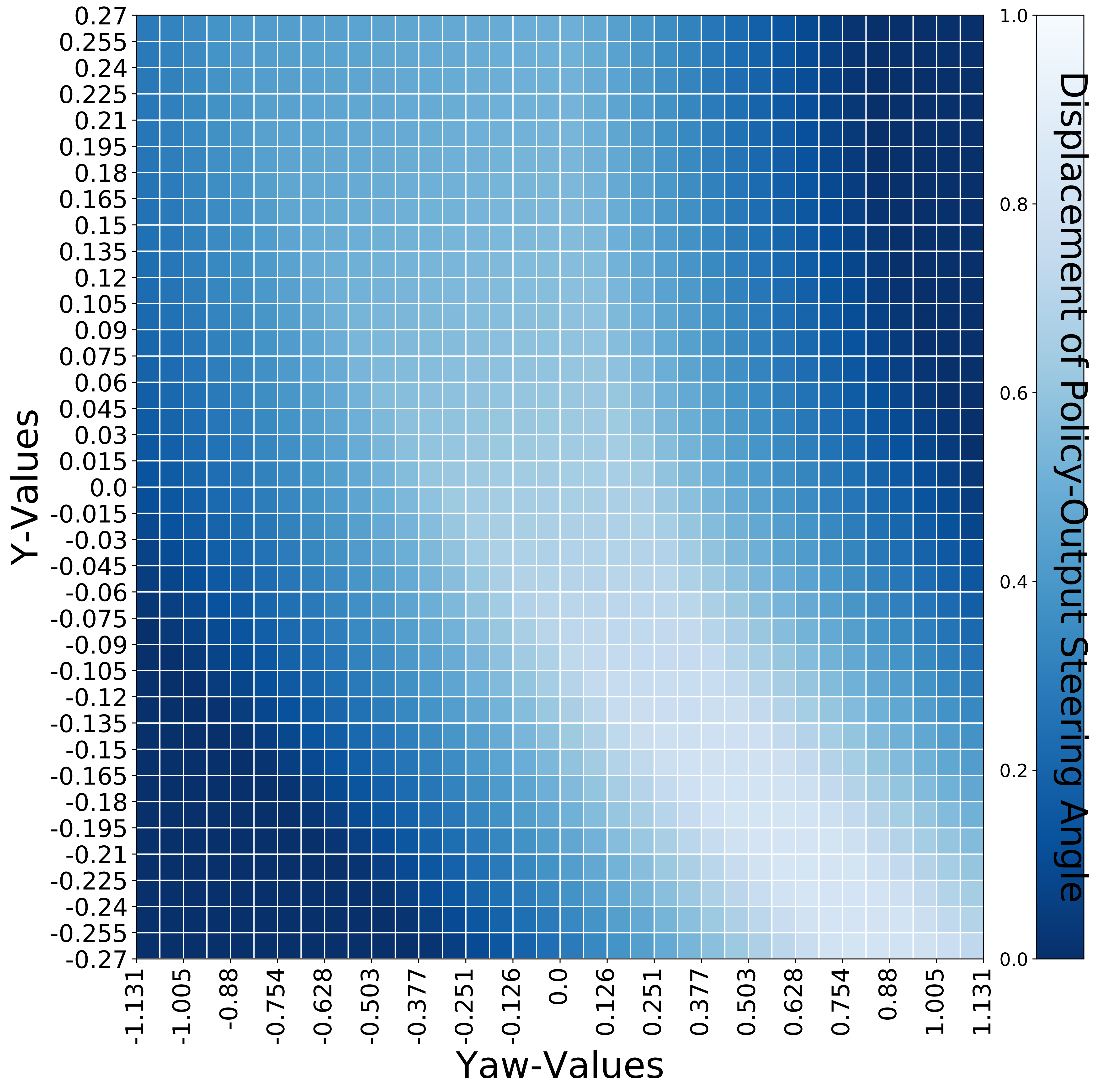




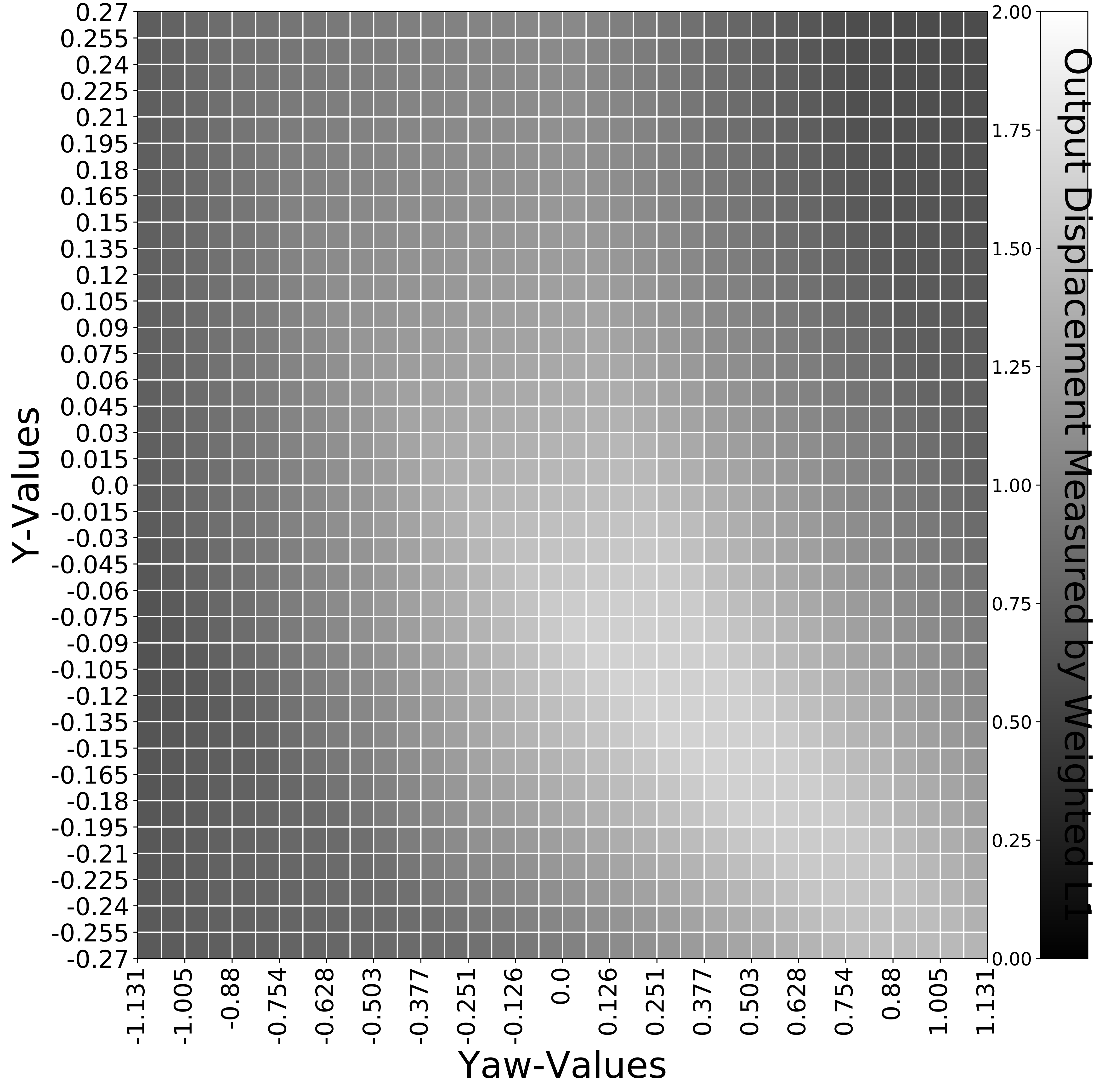
xDot:1.3, yDot:-0.3, yawDot: -1.5, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
4a76eb77-b4d2-45ef-b7d0-cb47ef9e6b5e

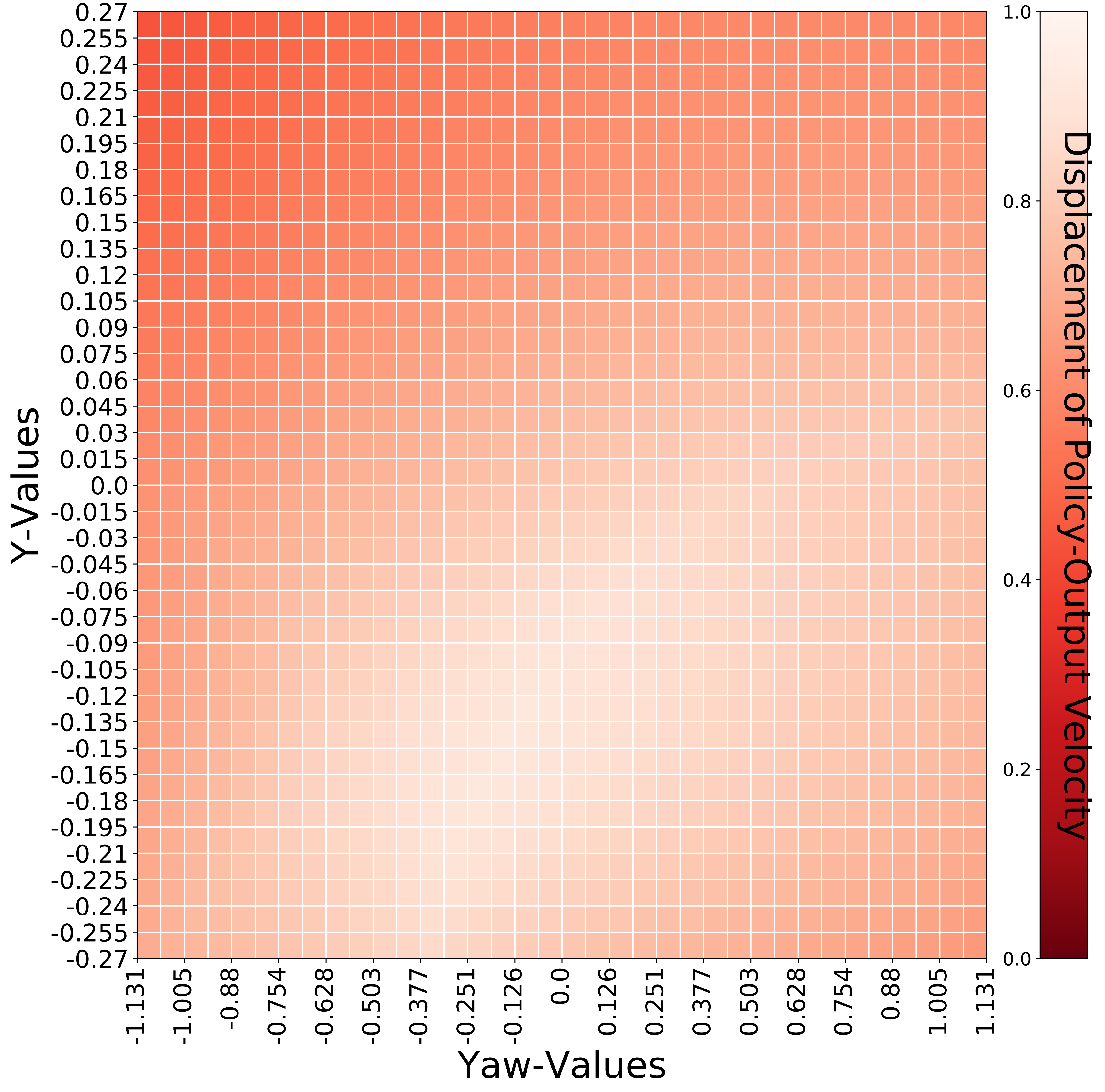


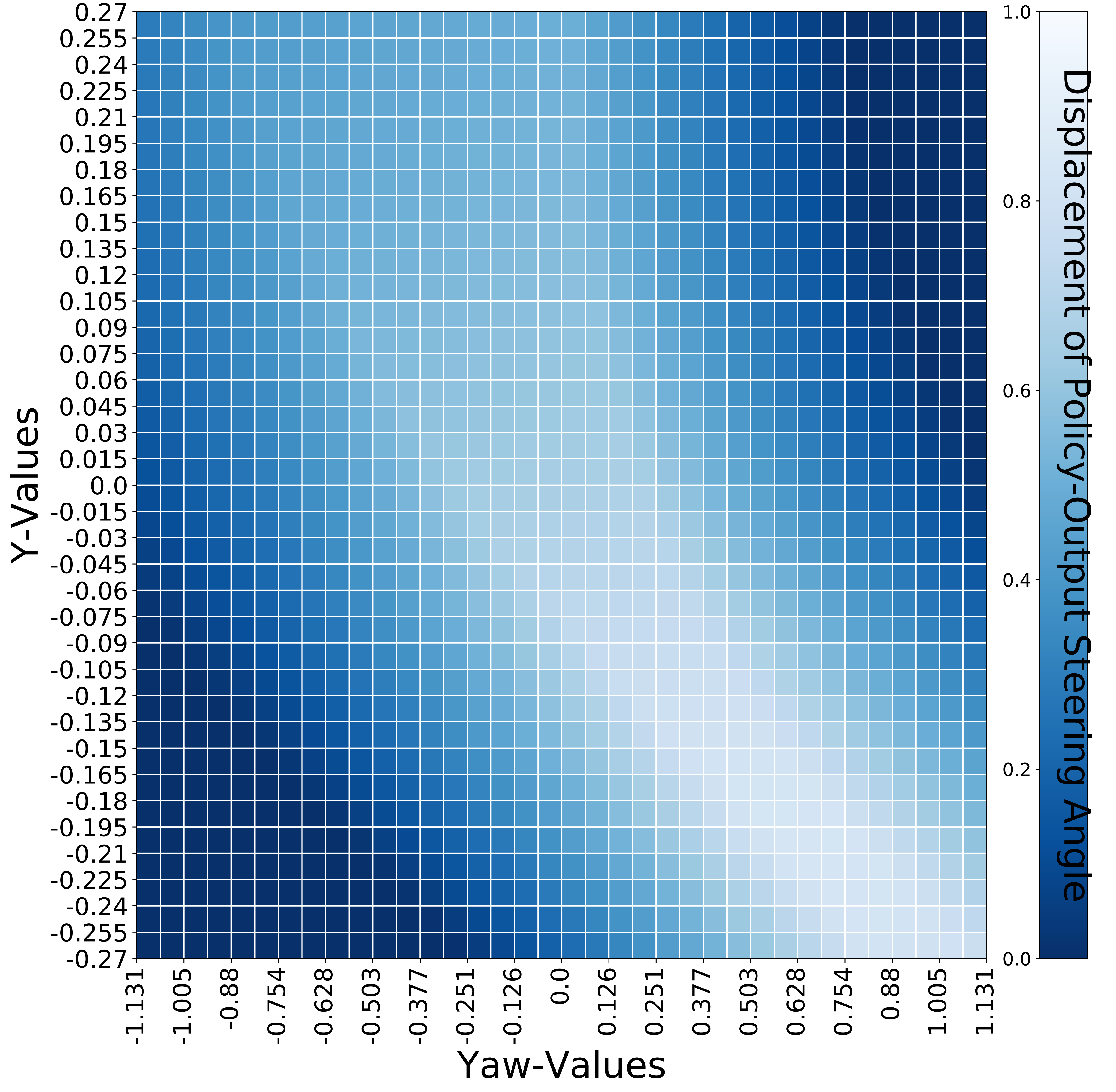




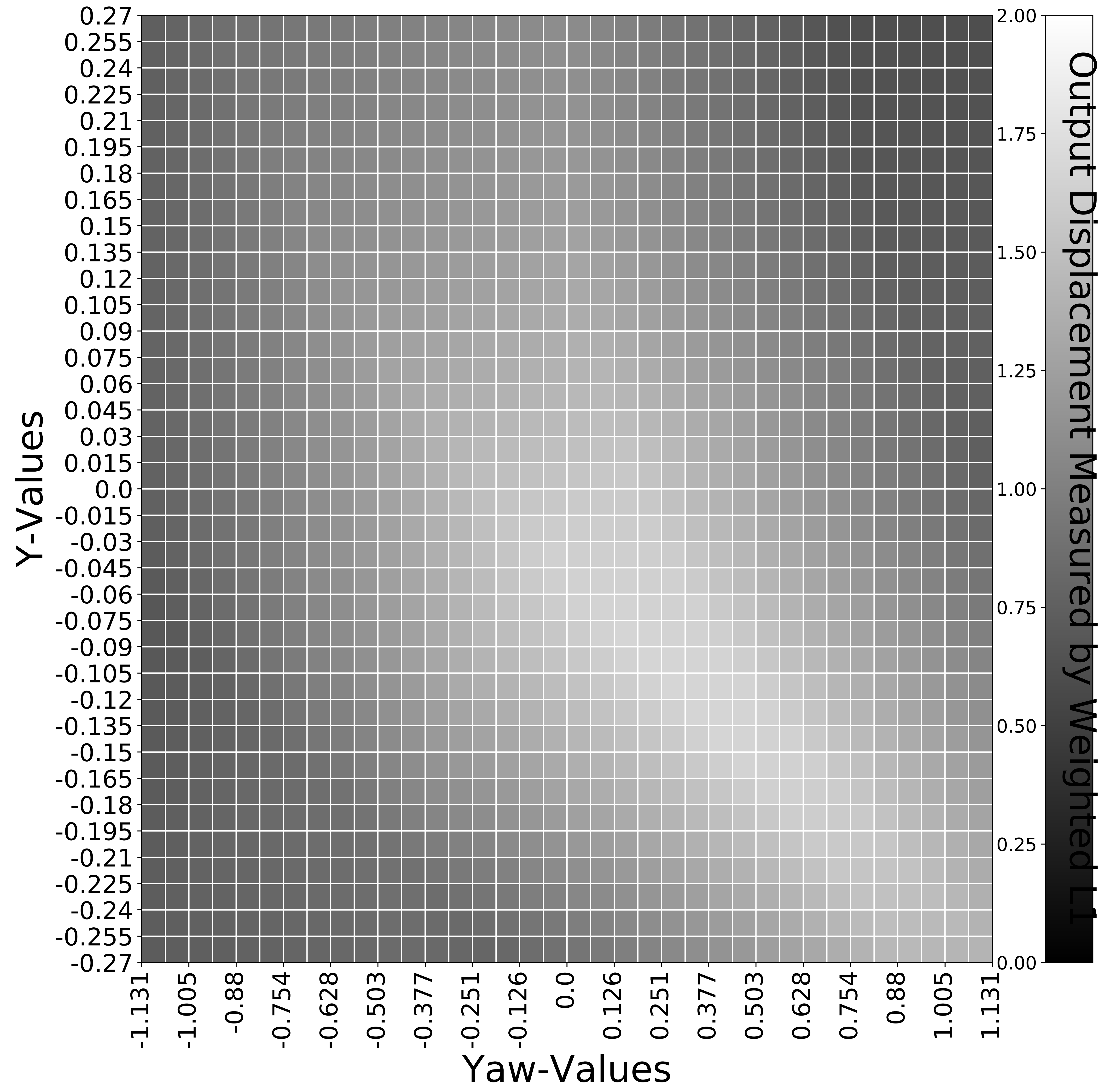
xDot:1.3, yDot:-0.6, yawDot: -1.25, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
827ade7f-3b47-4948-aedb-fc0fd7b62a5a

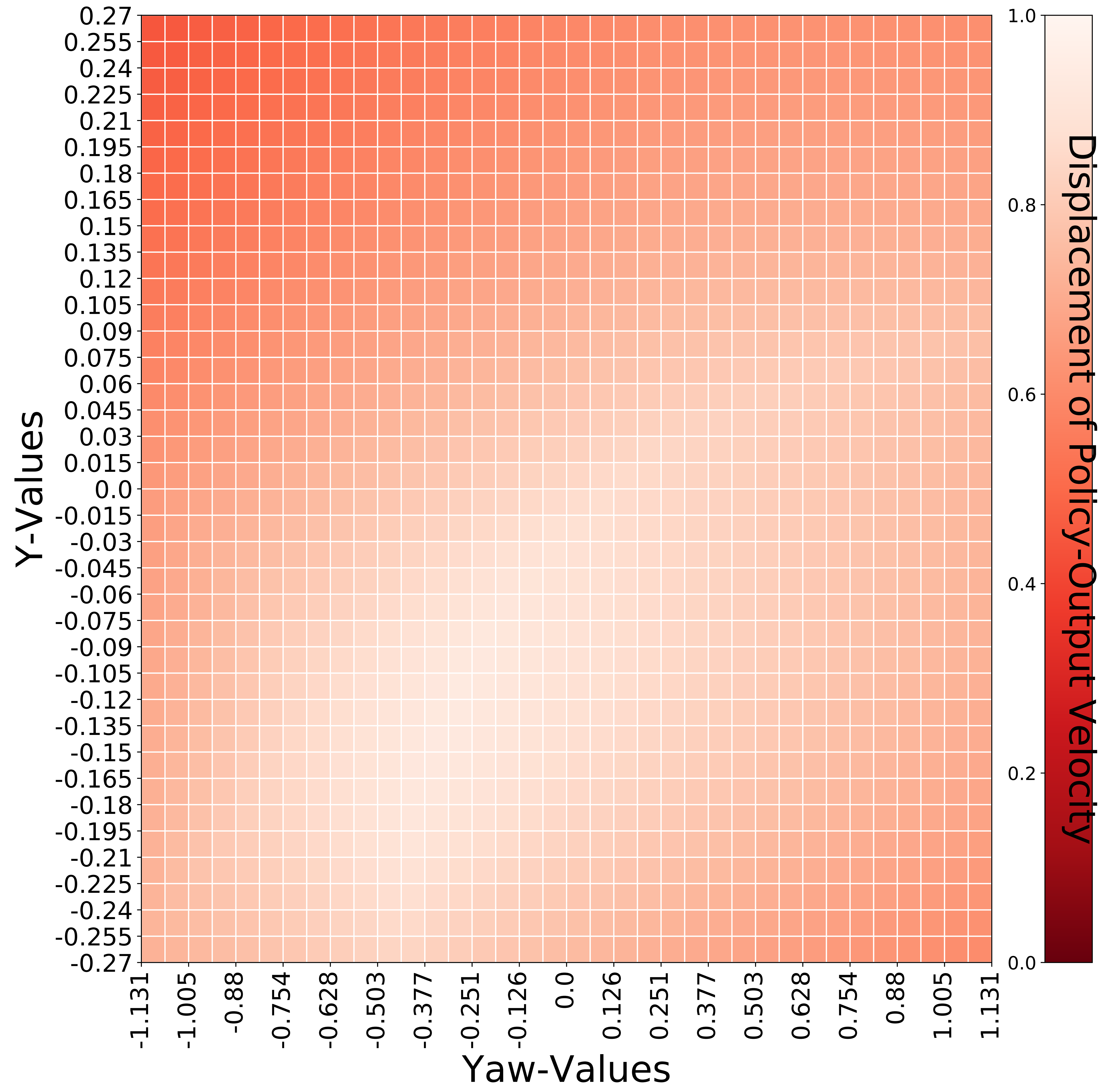


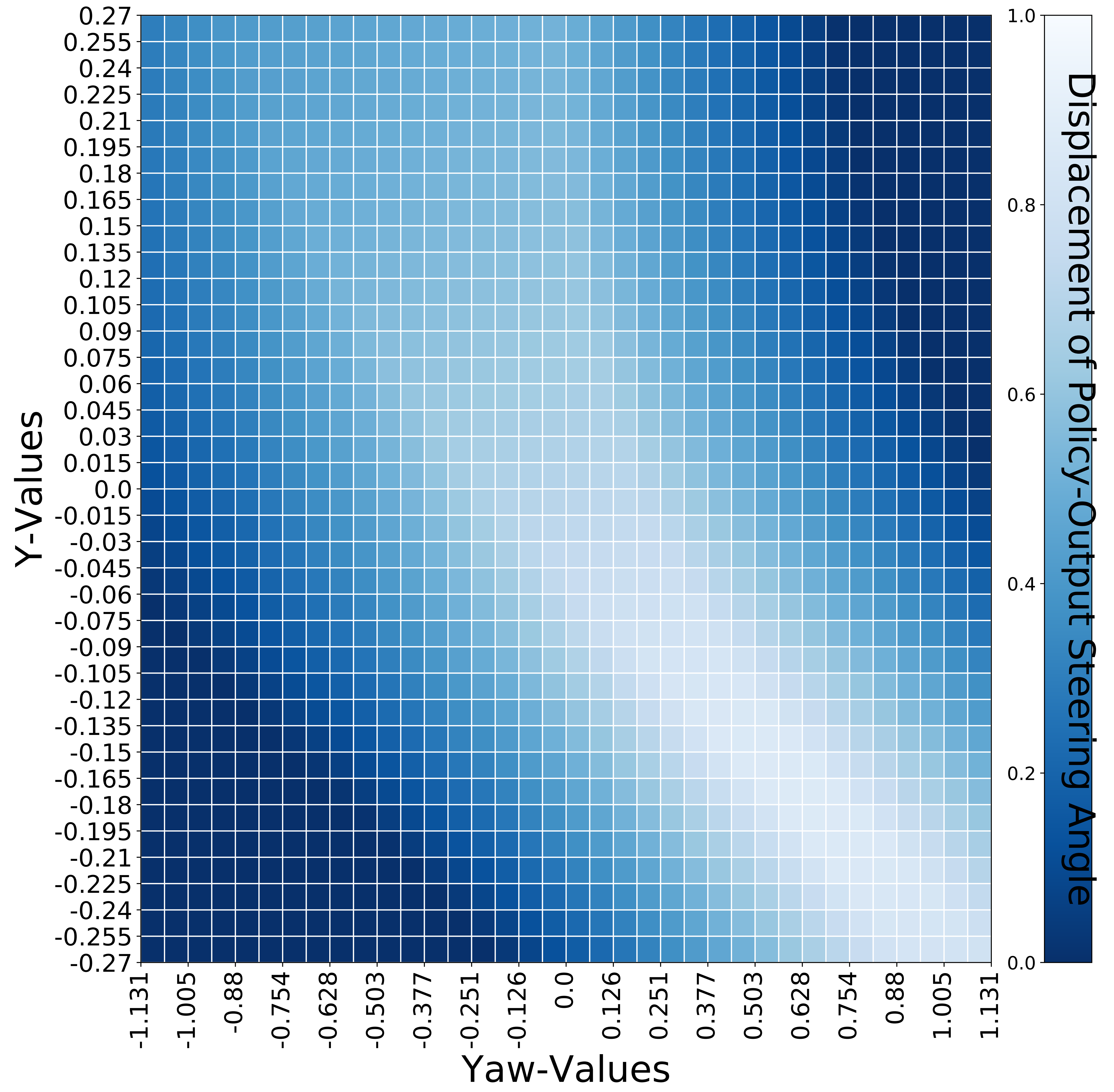




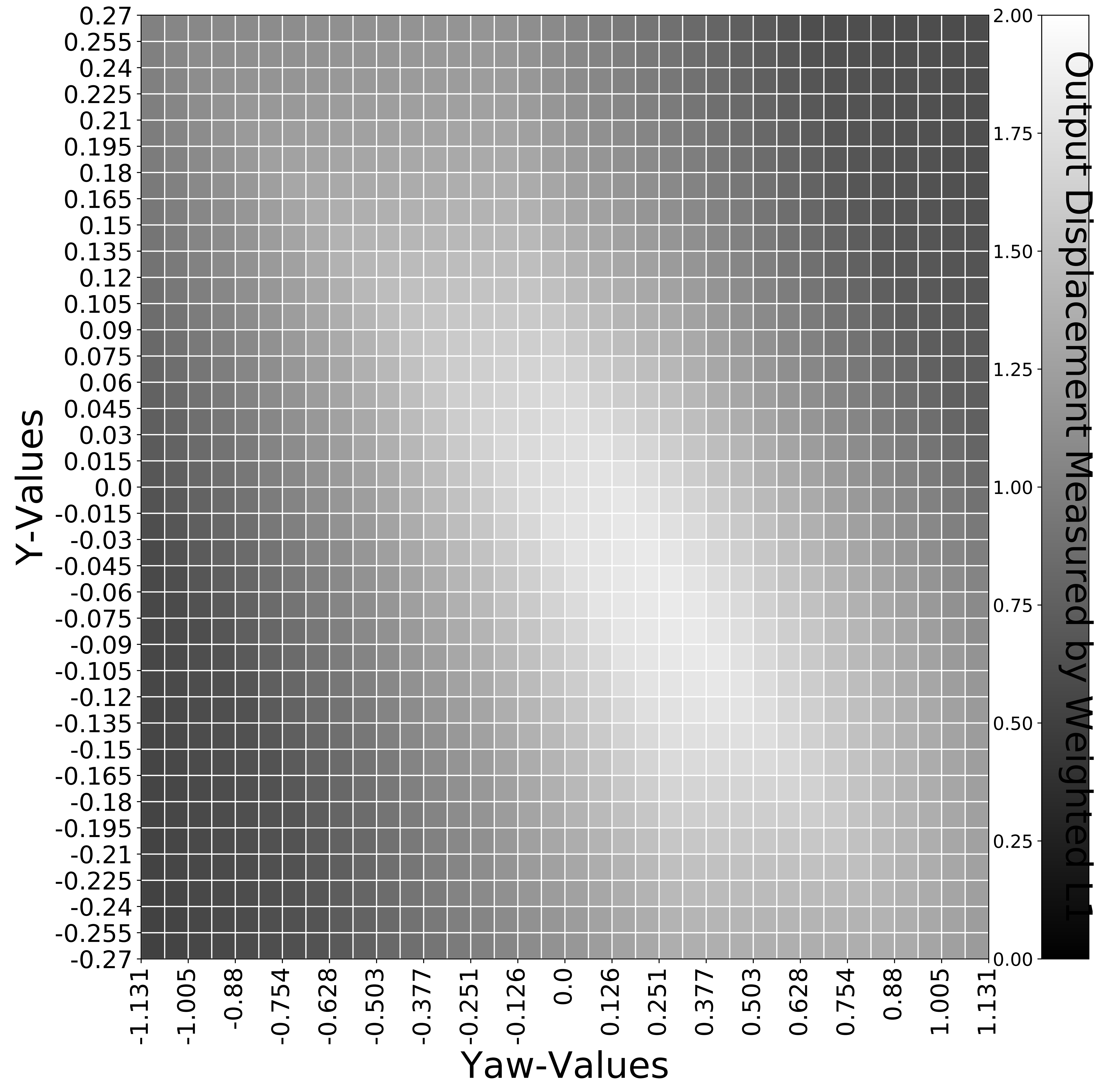
xDot:1.3, yDot:-0.6, yawDot: -0.75, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
a0e0c833-840e-44ab-a390-a32768fc9bfe

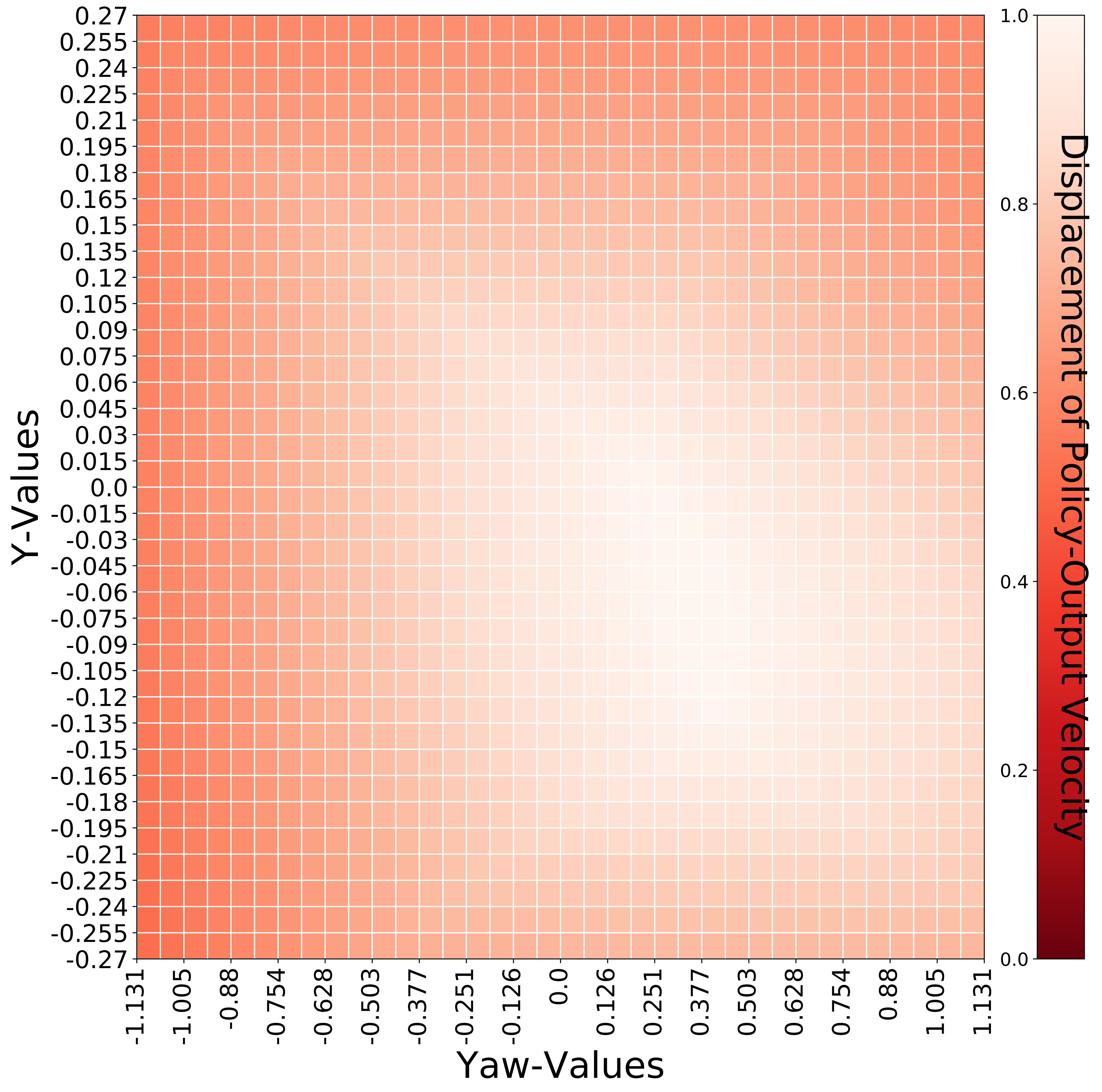


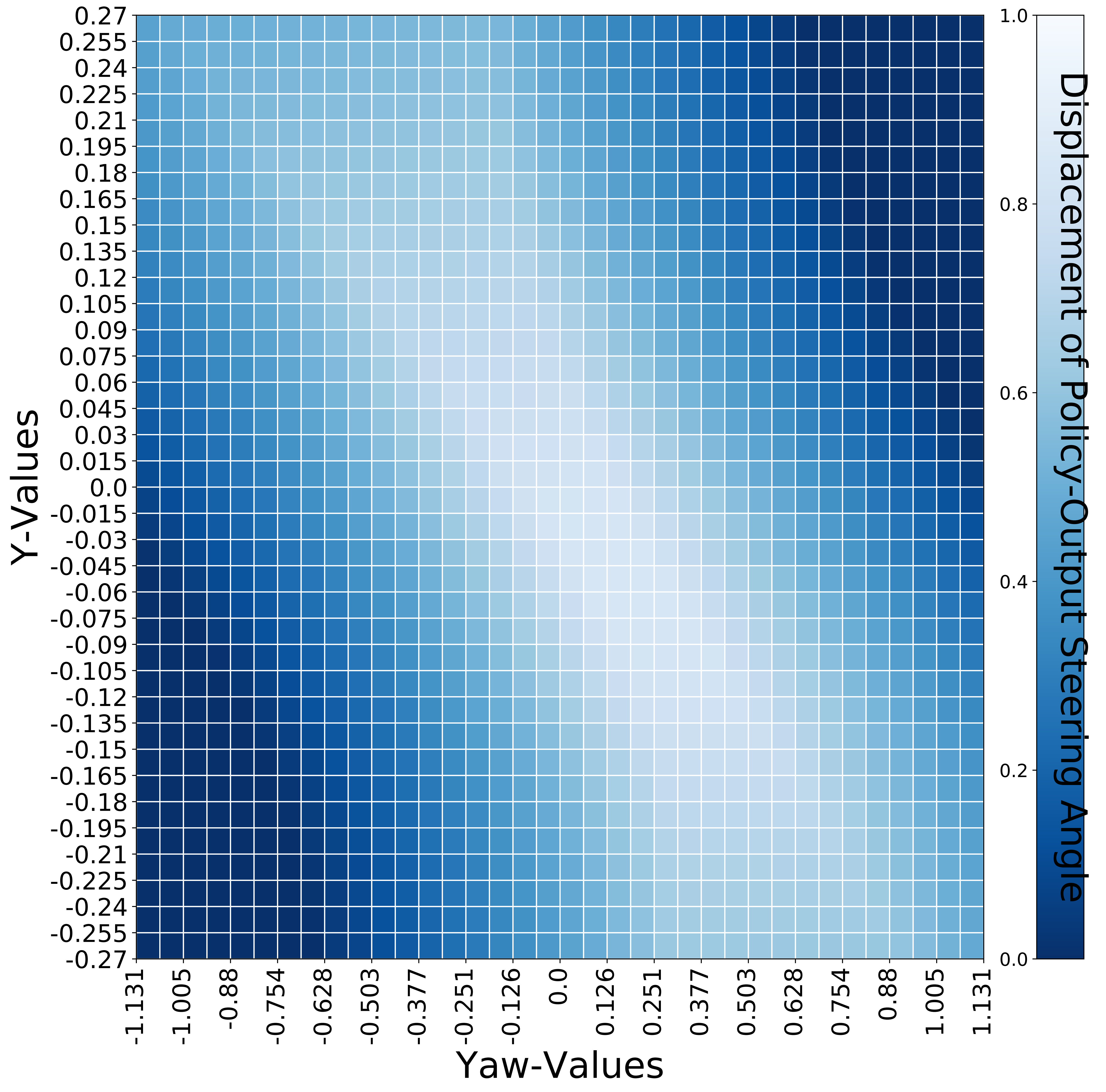




xDot:1.3, yDot:0.6, yawDot: 2.0, percentEpsilonToUse:0.015
Grey:displacement as L1-norm normalized per variable displayed
Red: displacement of policy-output velocity, Blue: displacement of policy-output steering angle
43aad7e1-e569-4c27-ba79-b89fed54a91e


Y-Values
Yaw-Values
Output Displacement Measured by Weighted L1


Y-Values
Yaw-Values
Displacement of Policy-Output Velocity


Y-Values
Yaw-Values
Displacement of Policy-Output Steering Angle