In [14]:
```
import tensorflow as tf
from tensorflow import keras

#import keras
from keras import layers
from keras import backend as K
from keras.models import Model
import numpy as np
img_shape = (28, 28, 1)
batch_size = 16
latent_dim = 2
input_img = keras.Input(shape=img_shape)
x = layers.Conv2D(32, 3,
    padding='same', activation='relu')(input_img)
x = layers.Conv2D(64, 3,
    padding='same', activation='relu',
    strides=(2, 2))(x)
x = layers.Conv2D(64, 3,
    padding='same', activation='relu')(x)
x = layers.Conv2D(64, 3,
    padding='same', activation='relu')(x)
shape_before_flattening = K.int_shape(x)
x = layers.Flatten()(x)
x = layers.Dense(32, activation='relu')(x)
z_mean = layers.Dense(latent_dim)(x)
z_log_var = layers.Dense(latent_dim)(x)
```

In [15]:
```
def sampling(args):
    z_mean, z_log_var = args
    epsilon = K.random_normal(shape=(K.shape(z_mean)[0], latent_dim),
        mean=0., stddev=1.)
    return z_mean + K.exp(z_log_var) * epsilon
z = layers.Lambda(sampling)([z_mean, z_log_var])
```

In [16]:
```
decoder_input = layers.Input(K.int_shape(z)[1:])
x = layers.Dense(np.prod(shape_before_flattening[1:]),
    activation='relu')(decoder_input)
x = layers.Reshape(shape_before_flattening[1:])(x)
x = layers.Conv2DTranspose(32, 3,
    padding='same',
    activation='relu',
    strides=(2, 2))(x)
x = layers.Conv2D(1, 3,
    padding='same',
    activation='sigmoid')(x)
decoder = Model(decoder_input, x)
z_decoded = decoder(z)
```

In [ ]:
```
```

In [17]:
```
class CustomVariationalLayer(keras.layers.Layer):

    def vae_loss(self, x, z_decoded):
        x = K.flatten(x)
        z_decoded = K.flatten(z_decoded)
        xent_loss = keras.metrics.binary_crossentropy(x, z_decoded)
        kl_loss = -5e-4 * K.mean(
            1 + z_log_var - K.square(z_mean) - K.exp(z_log_var), axis=-1)
        return K.mean(xent_loss + kl_loss)

    def call(self, inputs):
        x = inputs[0]
        z_decoded = inputs[1]
        loss = self.vae_loss(x, z_decoded)
        self.add_loss(loss, inputs=inputs)
        return x
y = CustomVariationalLayer()([input_img, z_decoded])
```

In [18]:
```
#import keras
import numpy as np

from keras import layers

import tensorflow.python.keras.backend as K
import tensorflow as tf
tf.compat.v1.disable_eager_execution()

from keras.datasets import mnist

from keras.models import Model

vae = Model(input_img, y)
vae.compile(optimizer='rmsprop', loss=None)
vae.summary()
(x_train, _), (x_test, y_test) = mnist.load_data()
x_train = x_train.astype('float32') / 255.
x_train = x_train.reshape(x_train.shape + (1,))
x_test = x_test.astype('float32') / 255.
x_test = x_test.reshape(x_test.shape + (1,))
vae.fit(x=x_train, y=None,
    shuffle=True,
    epochs=10,
    batch_size=batch_size,
    validation_data=(x_test, None))
```

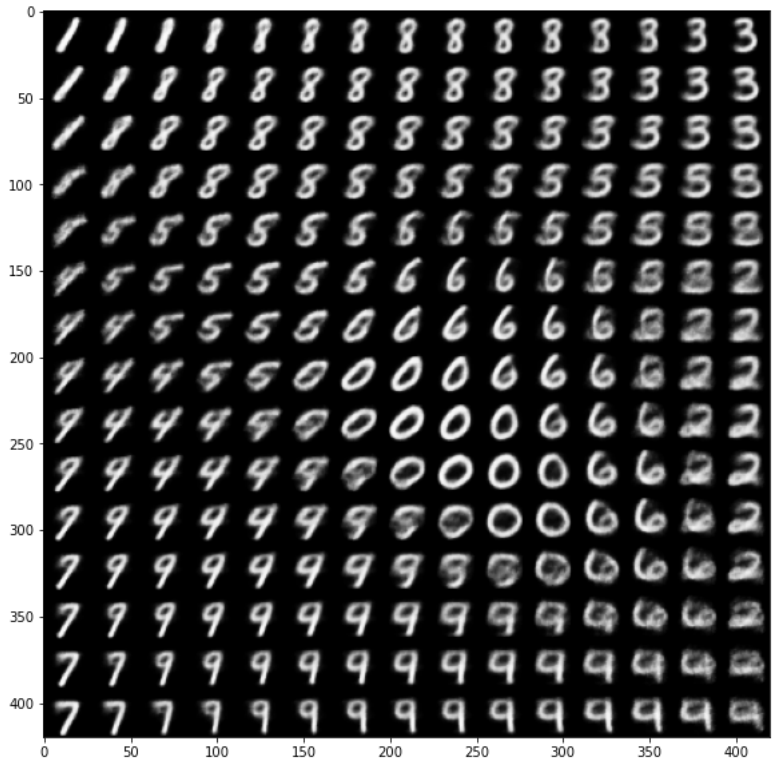
```
WARNING:tensorflow:Output custom_variational_layer_1 missing from loss dictionary. We assume this was done on purpose. The fit and evaluate APIs will not be expecting any data to be passed to custom_variational_layer_1.
Model: "model_9"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)           [(None, 28, 28, 1)]  0           []

 conv2d_5 (Conv2D)              (None, 28, 28, 32)   320         ['input_1[0][0]']

 conv2d_6 (Conv2D)              (None, 14, 14, 64)   18496       ['conv2d_5[0][0]']

 conv2d_7 (Conv2D)              (None, 14, 14, 64)   36928       ['conv2d_6[0][0]']

 conv2d_8 (Conv2D)              (None, 14, 14, 64)   36928       ['conv2d_7[0][0]']

 flatten_1 (Flatten)            (None, 12544)        0           ['conv2d_8[0][0]']

 dense_4 (Dense)                (None, 32)           401440      ['flatten_1[0][0]']

 dense_5 (Dense)                (None, 2)            66          ['dense_4[0][0]']

 dense_6 (Dense)                (None, 2)            66          ['dense_4[0][0]']

 lambda_1 (Lambda)              (None, 2)            0           ['dense_5[0][0]',
                                                                  'dense_6[0][0]']

 model_8 (Functional)           (None, 28, 28, 1)    56385       ['lambda_1[0][0]']

 custom_variational_layer_1 (Cu  (None, 28, 28, 1)   0           ['input_1[0][0]',
 stomVariationalLayer)                                            'model_8[0][0]']

==================================================================================================
Total params: 550,629
Trainable params: 550,629
Non-trainable params: 0
__________________________________________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
59984/60000 [============================>.] - ETA: 0s - loss: 0.2143

C:\ProgramData\Anaconda3\lib\site-packages\keras\engine\training_v1.py:2045: UserWarning: `Model.state_updates` will be removed in a future version. This property should not be used in TensorFlow 2.0, as `updates` are applied automatically.
  updates = self.state_updates
```

```
60000/60000 [==============================] - 149s 2ms/sample - loss: 0.2143 - val_loss: 0.1997
Epoch 2/10
60000/60000 [==============================] - 148s 2ms/sample - loss: 0.1957 - val_loss: 0.1929
Epoch 3/10
60000/60000 [==============================] - 1593s 27ms/sample - loss: 0.1913 - val_loss: 0.1902
Epoch 4/10
60000/60000 [==============================] - 129s 2ms/sample - loss: 0.1879 - val_loss: 0.1855
Epoch 5/10
60000/60000 [==============================] - 134s 2ms/sample - loss: 0.1855 - val_loss: 0.1854
Epoch 6/10
60000/60000 [==============================] - 128s 2ms/sample - loss: 0.1839 - val_loss: 0.1837
Epoch 7/10
60000/60000 [==============================] - 128s 2ms/sample - loss: 0.1827 - val_loss: 0.1842
Epoch 8/10
60000/60000 [==============================] - 132s 2ms/sample - loss: 0.1816 - val_loss: 0.1809
Epoch 9/10
60000/60000 [==============================] - 130s 2ms/sample - loss: 0.1808 - val_loss: 0.1801
Epoch 10/10
60000/60000 [==============================] - 131s 2ms/sample - loss: 0.1802 - val_loss: 0.1793
```

Out[18]: <keras.callbacks.History at 0x2066195d550>

In [19]:
```
import matplotlib.pyplot as plt
from scipy.stats import norm
n = 15
digit_size = 28
figure = np.zeros((digit_size * n, digit_size * n))
grid_x = norm.ppf(np.linspace(0.05, 0.95, n))
grid_y = norm.ppf(np.linspace(0.05, 0.95, n))
for i, yi in enumerate(grid_x):
    for j, xi in enumerate(grid_y):
        z_sample = np.array([[xi, yi]])
        z_sample = np.tile(z_sample, batch_size).reshape(batch_size, 2)
        x_decoded = decoder.predict(z_sample, batch_size=batch_size)
        digit = x_decoded[0].reshape(digit_size, digit_size)
        figure[i * digit_size: (i + 1) * digit_size,
            j * digit_size: (j + 1) * digit_size] = digit
plt.figure(figsize=(10, 10))
plt.imshow(figure, cmap='Greys_r')
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\keras\engine\training_v1.py:2067: UserWarning: `Model.state_updates` will be removed
in a future version. This property should not be used in TensorFlow 2.0, as `updates` are applied automatically.
  updates=self.state_updates,
```



In [12]:
```
print(tf.__version__)
```

```
2.9.1
```

In [ ]: