History, Trace, Loss investigates impermanence and mortality, emotional responses to death, and what might happen when we die. The works combine objects, photography and architectural space with projected video to evoke a sense of absent presence and the otherworldly. The dissertation situates the practice within art historical and contemporary art traditions - particularly superimposition and projection - and considers the relationship of visual technologies in conjuring otherworldly visions.

Keywords
projection; impermanence; ephemeral; time; mortality; interactive art; superimposition; spirit photography
for label, loss in zip(['Train', 'Validation'], ['loss', 'val_loss']): trace0 = {'type': 'scatter', 'x': df_history.index.tolist(), 'y': df_history[loss].tolist(), 'name': label, 'mode': 'lines'} trace.append(trace0) data = Data(trace).train_validation_loss(df_history). The model converged well as seen from the loss plot above. The next step is to use the model to identify outliers in new dataset. For this purpose I use the test data (X_test). Each of these tools is described in more detail below. Plotting History. The Keras fit() method returns an R object containing the training history, including the value of metrics at the end of each epoch. You can plot the training metrics by epoch using the plot() method. For example, here we compile and fit a model with the "accuracy" metric: model %>% compile(loss = 'categorical_crossentropy', optimizer = optimizer_rmsprop(), metrics = c('accuracy')).