



UNIVERSITÀ DI PARMA

Dipartimento di Ingegneria e Architettura

Corso di Laurea in Ingegneria Informatica - Informatica Industriale

Deep Marker Detector

Vincenzo Fraello (339641) - Filippo D'Addeo (339643)

ANNO ACCADEMICO 2022/2023

1 Descrizione progetto

L'obiettivo principale di questo progetto è quello di sviluppare un sistema basato sul *Deep Learning* (DL) per l'estrazione di marker circolari da immagini bidimensionali utilizzate per la calibrazione delle telecamere.

Le immagini utilizzate per l'addestramento del modello sono di due tipologie, e variano in relazione ai parametri risoluzione-obiettivo:

- 1920×1080 pinhole (vedi Figura 1);
- 3840×1920 fisheye (vedi Figura 2);

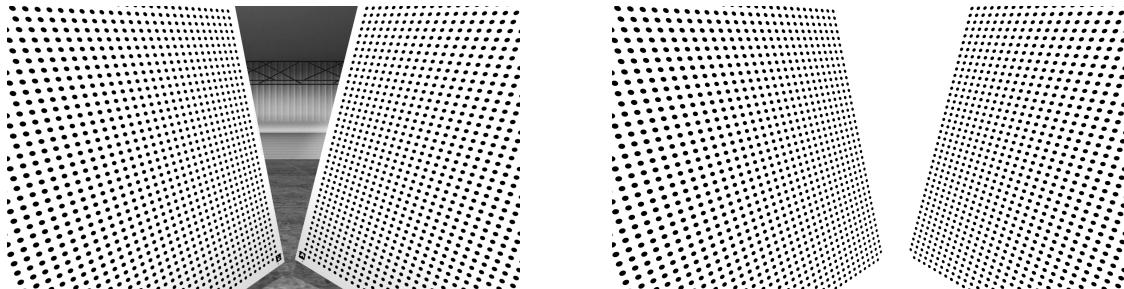


Figura 1: Immagine bidimensionale per la calibrazione e corrispettiva maschera binaria

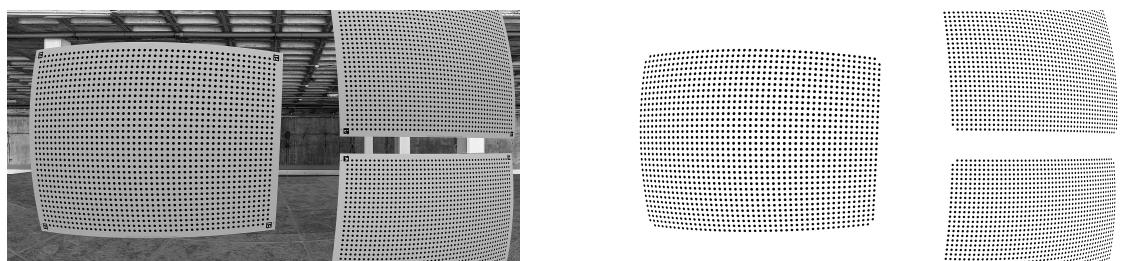


Figura 2: Immagine bidimensionale per la calibrazione e corrispettiva maschera binaria

2 Strumenti utilizzati

Il linguaggio di programmazione utilizzato per lo sviluppo del sistema è *Python*, che ha permesso l'uso di *PyTorch*: framework end-to-end per il *Machine Learning* (ML) utilizzato per lo sviluppo, l'addestramento e la validazione del modello di DL.

Nello specifico, i modelli utilizzati sono i *Convolutional AutoEncoder* (CAE): architetture di DL composte da strati convoluzionali per la creazione di rappresentazioni compresse delle immagini di input. Tra gli altri usi, questi modelli vengono tipicamente impiegati per i task di *Image Compression*, *Image Generation*, *Feature Extraction* e *Semantic Segmentation*.

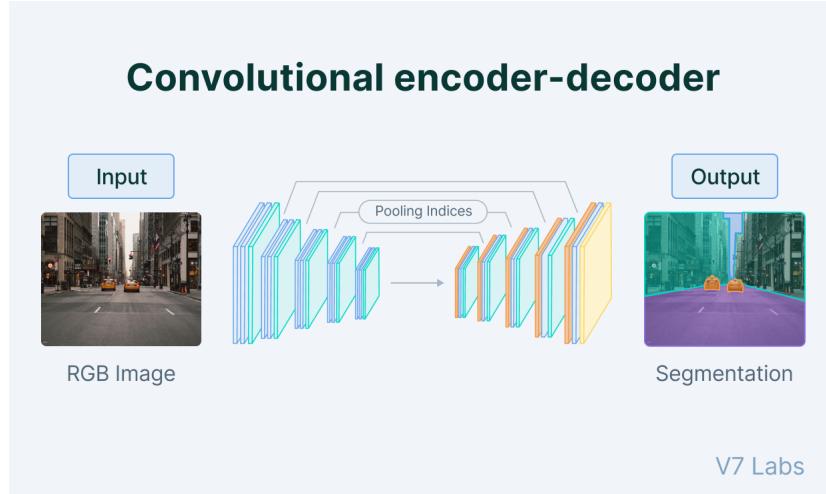


Figura 3: Architettura generale CAE

Le Figure 4 e 5 mostrano i dettagli implementativi delle due architetture utilizzate per il task di estrazione dei marker circolari. I due modelli si distinguono per la loro capacità: il modello con i layer *ConvTranspose2d* ha più parametri perché, a differenza dell'*Upsample*, questi layer contengono parametri apprendibili.

Layer (type)	Output Shape	Param #	Tr. Param #
<hr/>			
Conv2d-1	[2, 64, 1080, 1920]	640	640
ReLU-2	[2, 64, 1080, 1920]	0	0
MaxPool2d-3	[2, 64, 1079, 1919]	0	0
Conv2d-4	[2, 16, 1079, 1919]	9,232	9,232
ReLU-5	[2, 16, 1079, 1919]	0	0
MaxPool2d-6	[2, 16, 1078, 1918]	0	0
Upsample-7	[2, 16, 1078, 1918]	0	0
Conv2d-8	[2, 64, 1078, 1918]	9,280	9,280
ReLU-9	[2, 64, 1078, 1918]	0	0
Upsample-10	[2, 64, 1078, 1918]	0	0
Conv2d-11	[2, 1, 1080, 1920]	577	577
Sigmoid-12	[2, 1, 1080, 1920]	0	0
<hr/>			
Total params: 19,729			
Trainable params: 19,729			
Non-trainable params: 0			
<hr/>			

Figura 4: Architettura CAE con *Upsample* layers

Layer (type)	Output Shape	Param #	Tr. Param #
Conv2d-1	[2, 16, 540, 960]	256	256
BatchNorm2d-2	[2, 16, 540, 960]	32	32
ReLU-3	[2, 16, 540, 960]	0	0
Conv2d-4	[2, 32, 270, 480]	8,192	8,192
BatchNorm2d-5	[2, 32, 270, 480]	64	64
ReLU-6	[2, 32, 270, 480]	0	0
Conv2d-7	[2, 64, 135, 240]	32,832	32,832
ConvTranspose2d-8	[2, 32, 270, 480]	8,224	8,224
BatchNorm2d-9	[2, 32, 270, 480]	64	64
ReLU-10	[2, 32, 270, 480]	0	0
ConvTranspose2d-11	[2, 16, 540, 960]	2,064	2,064
BatchNorm2d-12	[2, 16, 540, 960]	32	32
ReLU-13	[2, 16, 540, 960]	0	0
ConvTranspose2d-14	[2, 1, 1080, 1920]	65	65
Sigmoid-15	[2, 1, 1080, 1920]	0	0
<hr/>			
Total params: 51,825			
Trainable params: 51,825			
Non-trainable params: 0			
<hr/>			

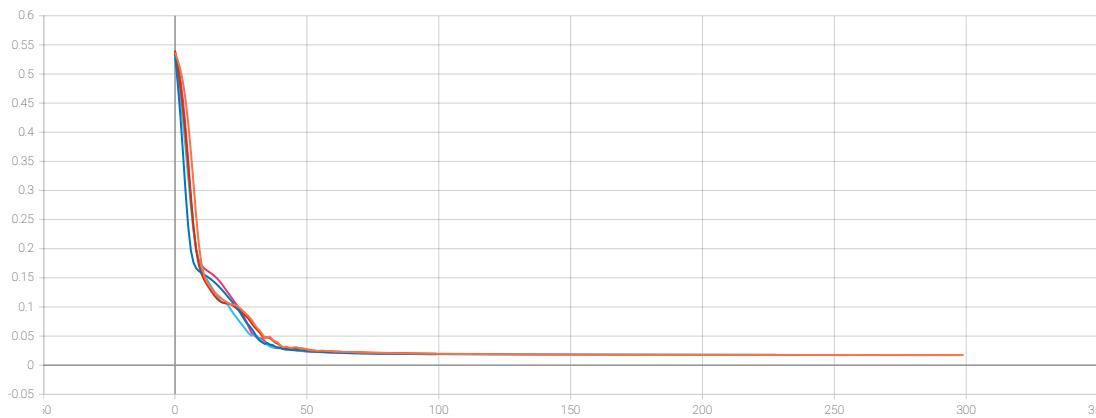
Figura 5: Architettura CAE con *ConvTranspose2d* layers

3 Risultati

I test sono stati eseguiti al variare delle *loss-function* (*BinaryCrossEntropyWithLogitsLoss*, *IntersectionOverUnionLoss*, *DiceLoss*)¹, del *random-seed* (da 1 a 5) per la composizione del dataset, del *batch-size*, della normalizzazione e delle trasformazioni applicate al *training-set*.

CAE-Upsample 1920×1080-pinhole configurazione di test che ha previsto l’uso di:

- CAE con Upsample layer;
- Dataset composto dalle sole immagini 1920×1080 pinhole;
- Loss-function: *IntersectionOverUnionLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 2;

Figura 6: Training loss $\times 5$ run al variare del random-seed da 1 a 5

¹La Dice-Loss è più efficace quando le classi di oggetti da segmentare sono sbilanciate o quando si desidera massimizzare la sovrapposizione tra la maschera di segmentazione prevista e quella reale. La IOU-Loss (JaccardIndex-Loss) è più adatta quando l’obiettivo principale è la segmentazione precisa delle classi di oggetti, piuttosto che la sovrapposizione con la maschera di segmentazione reale.

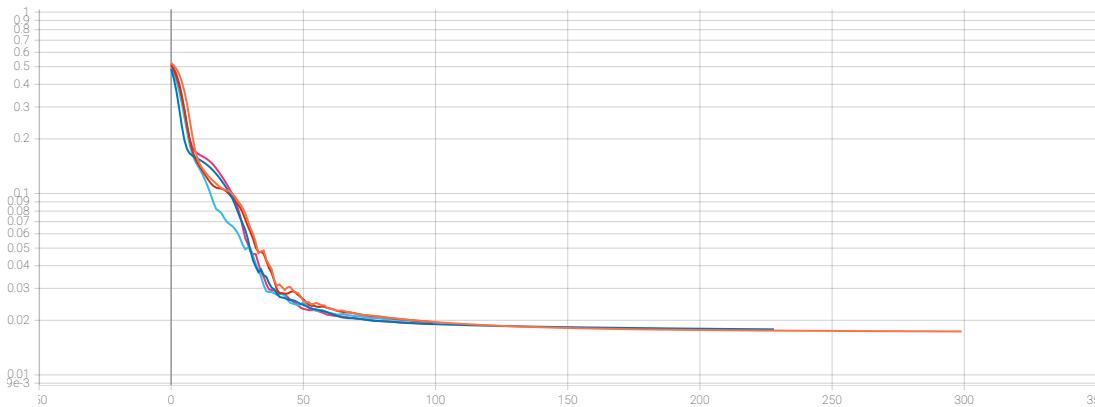
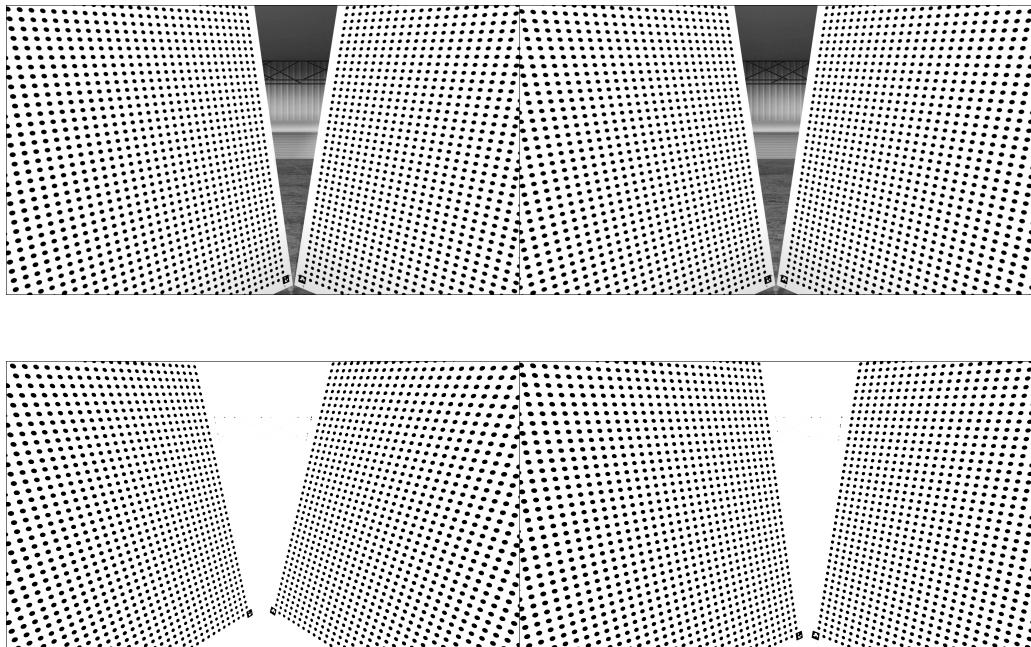
Figura 7: Validation loss $\times 5$ run al variare del random-seed da 1 a 5

Figura 8: Risultati qualitativi estrazione marker

Osservazioni: dai grafici e dalle immagini sopra è possibile osservare che: (i) al variare della composizione del dataset il modello è stabile; (ii) l'errore in validation è basso a causa dell'elevata similarità tra immagini di training e immagini di validation; (iii) la rete impara a reagire al marker nero, quindi viene stimolata dalle bande nere presenti all'interno dell'immagine, e queste vengono ricostruite anche nella maschera di output.

CAE-Upsample 1920×1080-pinhole configurazione di test che ha previsto l'uso di:

- CAE con Upsample layer;
- Dataset composto dalle sole immagini 1920×1080 pinhole;
- Loss-function: *DiceLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 2;

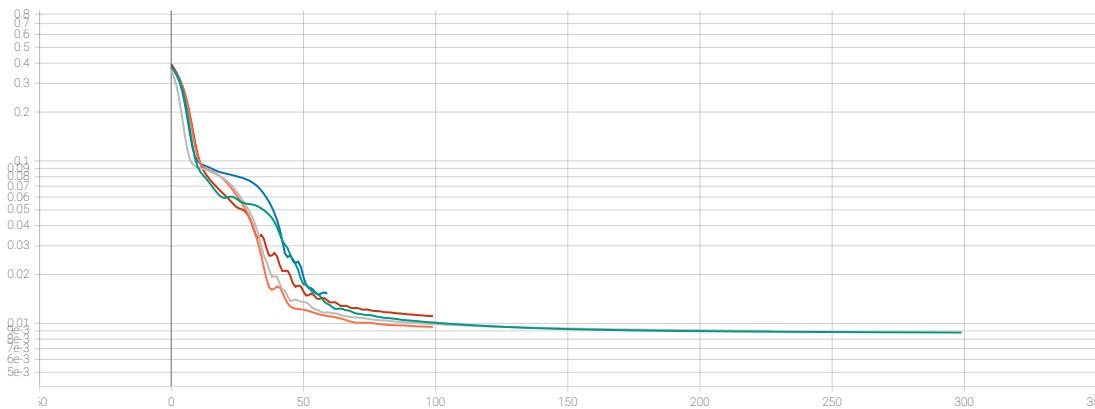


Figura 9: Training loss × 5 run al variare del random-seed da 1 a 5

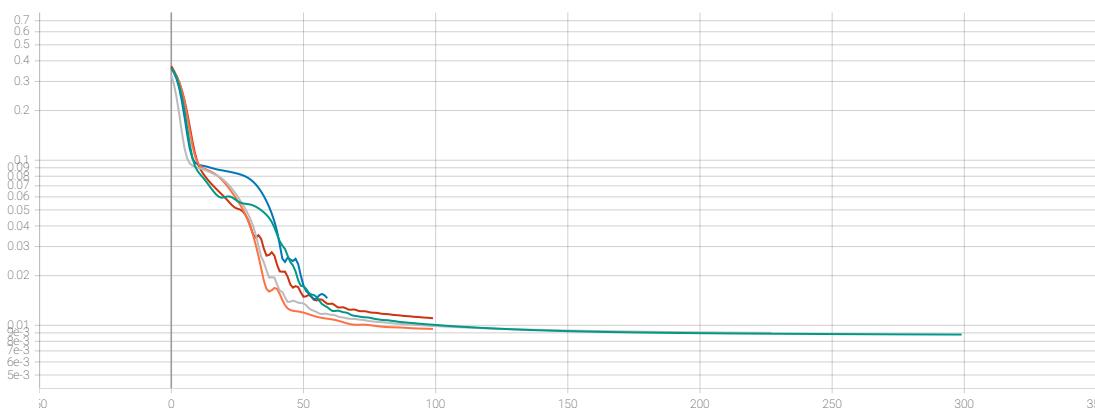


Figura 10: Validation loss × 5 run al variare del random-seed da 1 a 5

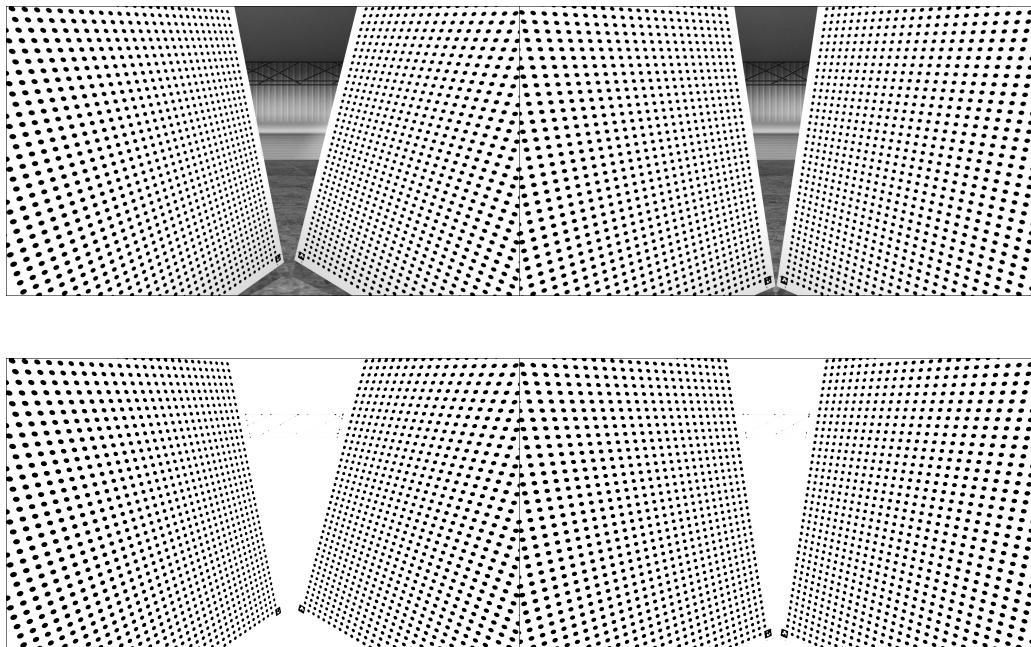


Figura 11: Risultati qualitativi estrazione marker

Osservazioni: dai grafici e dalle immagini sopra è possibile osservare che: (i) al variare della composizione del dataset il modello è stabile; (ii) l'errore in validation è basso a causa dell'elevata similarità tra immagini di training e immagini di validation; (iii) la rete impara a reagire al marker nero, quindi viene stimolata dalle bande nere presenti all'interno dell'immagine, e queste vengono ricostruite anche nella maschera di output.

CAE-ConvTranspose 1920×1080-pinhole configurazione di test che ha previsto l'uso di:

- CAE con ConvTranspose2d layer;
- Dataset composto dalle sole immagini 1920×1080 pinhole;
- Loss-function: *IntersectionOverUnionLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 2;

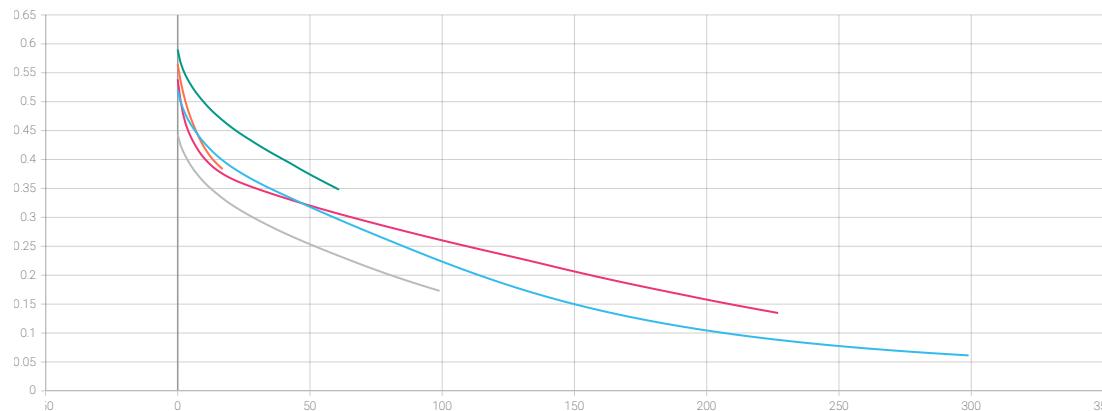


Figura 12: Training loss $\times 5$ run al variare del random-seed da 1 a 5

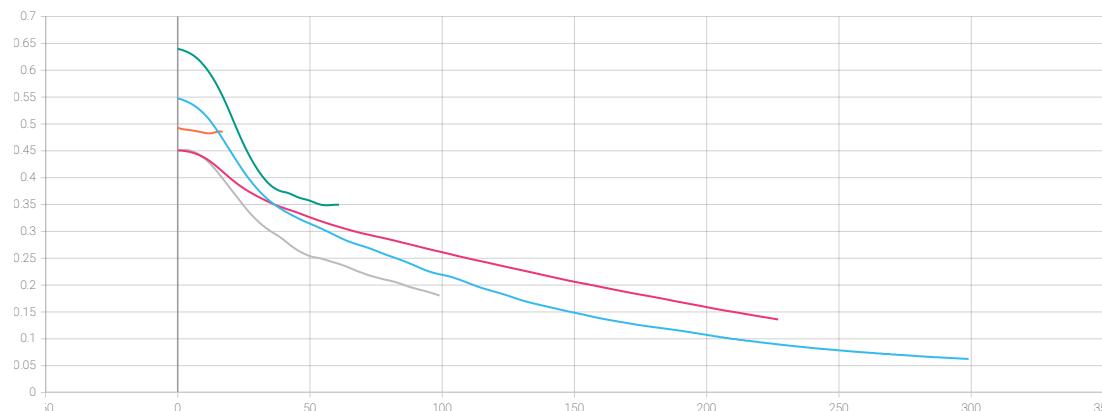
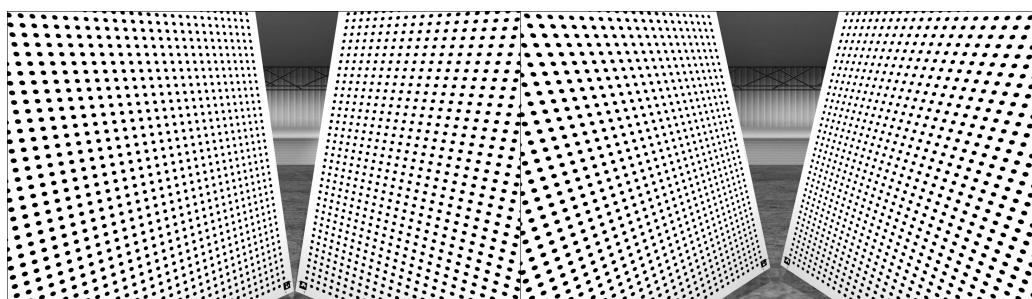


Figura 13: Validation loss $\times 5$ run al variare del random-seed da 1 a 5



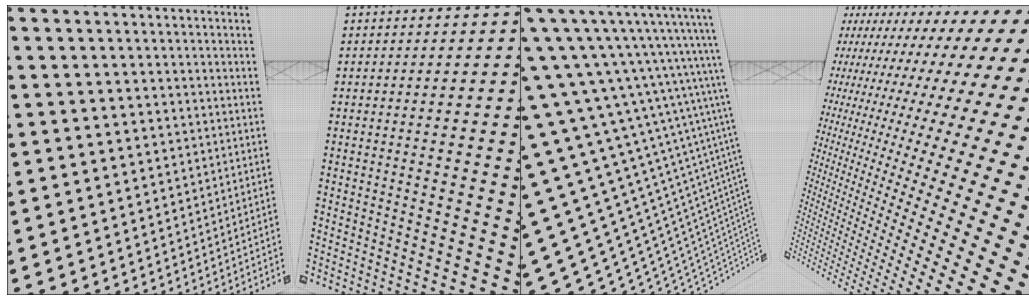
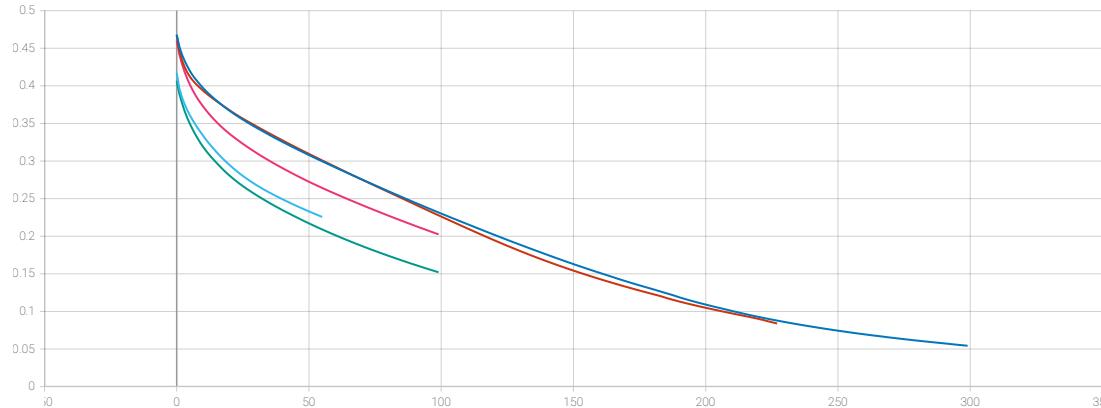
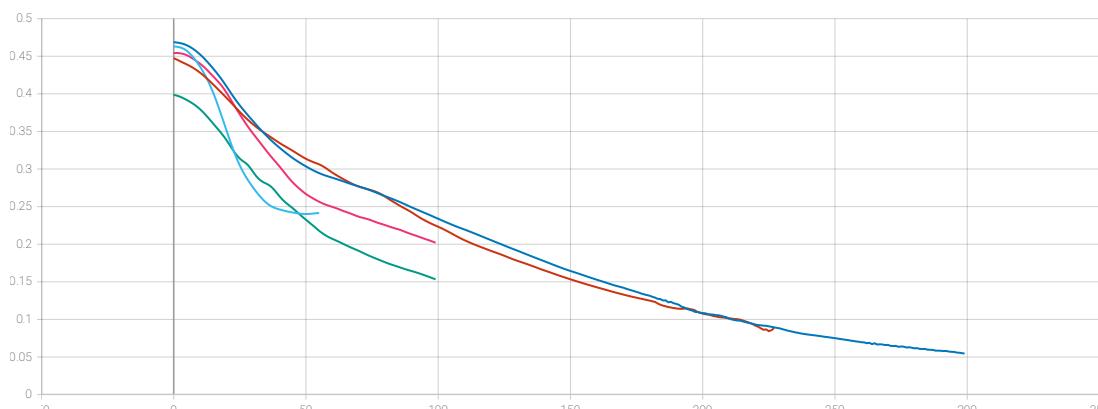


Figura 14: Risultati qualitativi estrazione marker

Osservazioni: dai grafici e dalle immagini sopra è possibile osservare che: (i) al variare della composizione del dataset il modello non è stabile: ci sono configurazioni del training-set che portano a risultati qualitativi scadenti.

CAE-ConvTranspose 1920×1080-pinhole configurazione di test che ha previsto l'uso di:

- CAE con ConvTranspose2d layer;
- Dataset composto dalle sole immagini 1920×1080 pinhole;
- Loss-function: *DiceLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 2;

Figura 15: Training loss $\times 5$ run al variare del random-seed da 1 a 5Figura 16: Validation loss $\times 5$ run al variare del random-seed da 1 a 5

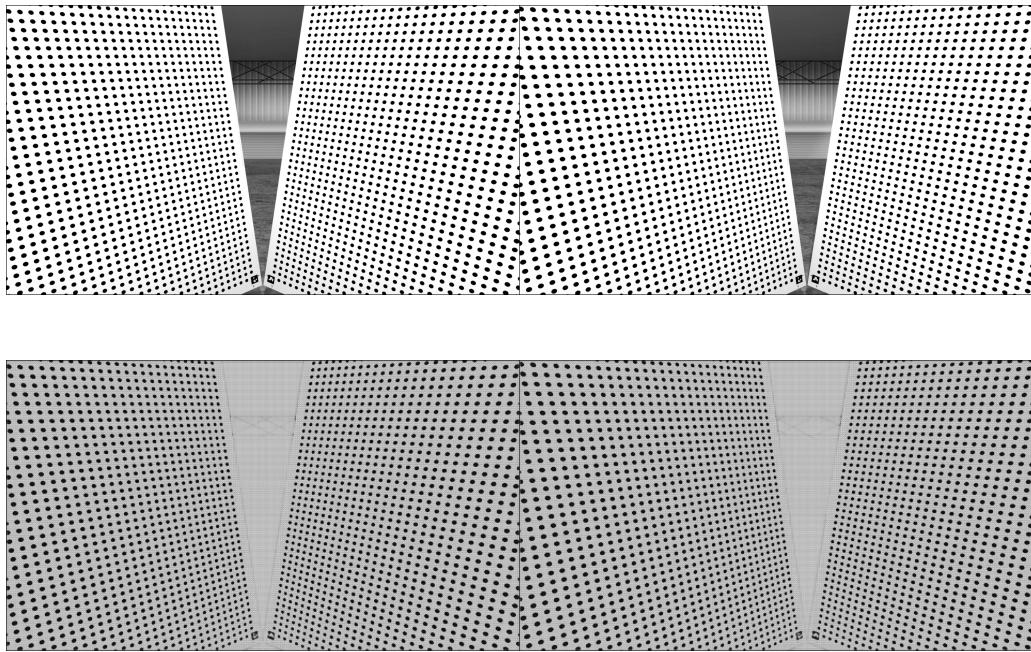
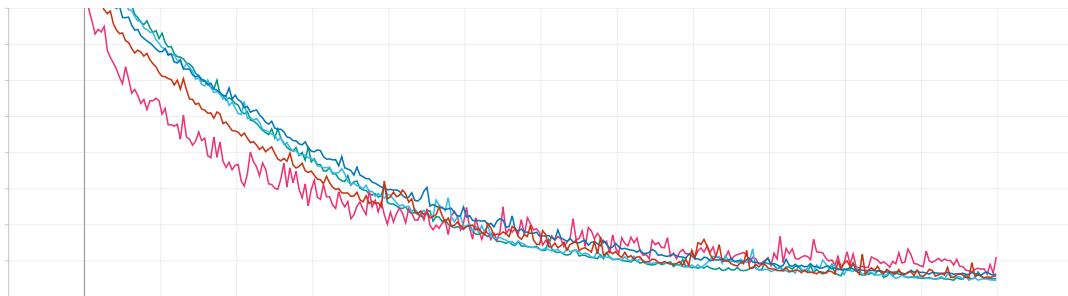


Figura 17: Risultati qualitativi estrazione marker

Osservazioni: dai grafici e dalle immagini sopra è possibile osservare che: (i) al variare della composizione del dataset il modello non è stabile: ci sono configurazioni del training-set che portano a risultati qualitativi scadenti.

CAE-ConvTranspose 3840×1920-fisheye configurazione di test che ha previsto l'uso di:

- CAE con ConvTranspose2d layer;
- Dataset composto dalle sole immagini 3840×1920 fisheye;
- Loss-function: *IntersectionOverUnionLoss*;
- No Normalizzazione;
- Trasformazioni: abilitate;
- Batch-size: 4;

Figura 18: Training loss $\times 5$ run al variare del random-seed da 1 a 5

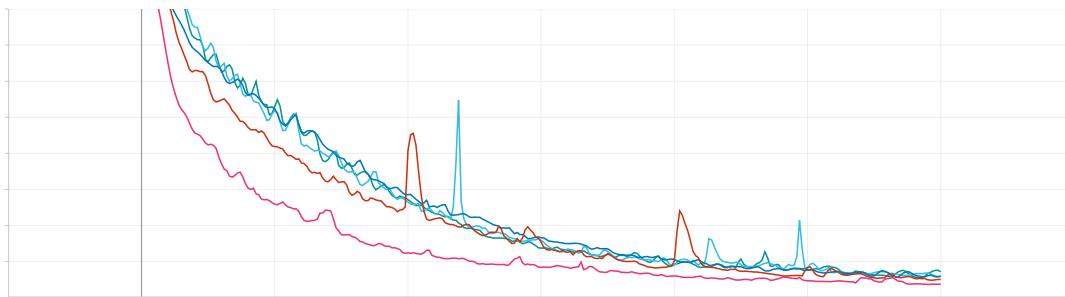
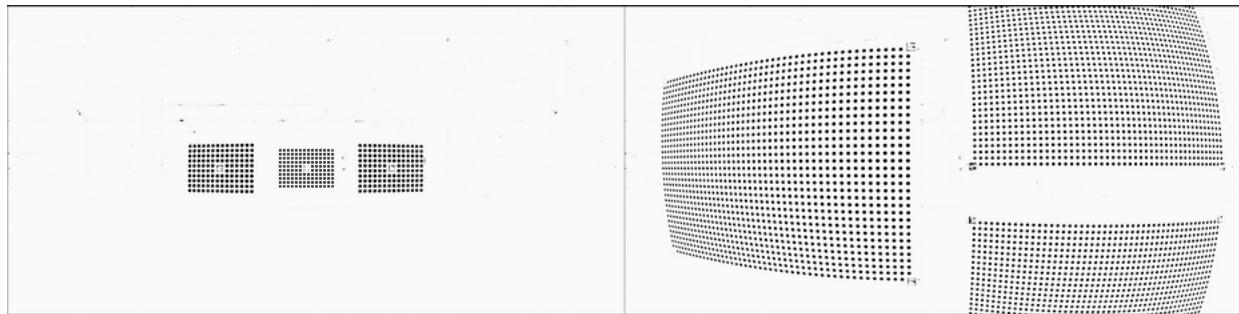


Figura 19: Validation loss $\times 5$ run al variare del random-seed da 1 a 5



Osservazioni: dai grafici e dalle immagini sopra è possibile osservare che, diversamente ad quanto mostrato con le immagini pin-hole, il modello è in grado di comportarsi nel modo corretto, evidenziando però qualche difficoltà con i marker Aruco. In particolare, questa correttezza maggiore può essere ricondotta all'utilizzo di un batch size maggiore.

CAE-ConvTranspose 3840×1920-fisheye configurazione di test che ha previsto l'uso di:

- CAE con ConvTranspose2d layer;
- Dataset composto dalle sole immagini 3840×1920 fisheye;
- Loss-function: *DiceLoss*;
- No Normalizzazione;
- Trasformazioni: abilitate;
- Batch-size: 4;



Figura 20: Training loss.

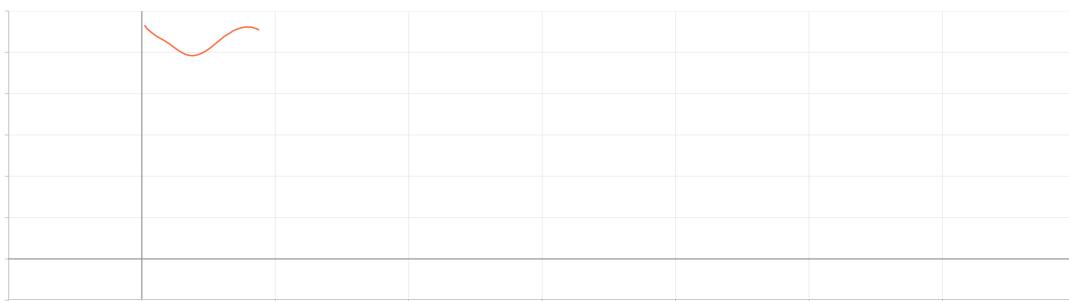


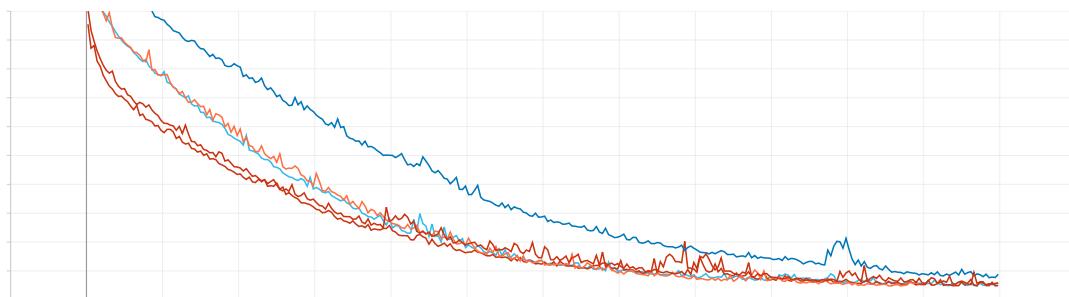
Figura 21: Validation loss.



Osservazioni: dai grafici e dalle immagini sopra è possibile osservare che il modello non è assolutamente in grado di apprendere il pattern desiderato, evidenziando una loss di training molto elevata e una loss di validation molto instabile. Per questo motivo, non sono stati eseguiti ulteriori test con la *DiceLoss* nell'ambito delle fisheye.

CAE-ConvTranspose 3840×1920-fisheye configurazione di test che ha previsto l'uso di:

- CAE con ConvTranspose2d layer;
- Dataset composto dalle sole immagini 3840×1920 fisheye;
- Loss-function: *IntersectionOverUnionLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 4;

Figura 22: Training loss $\times 5$ run al variare del random-seed da 1 a 5

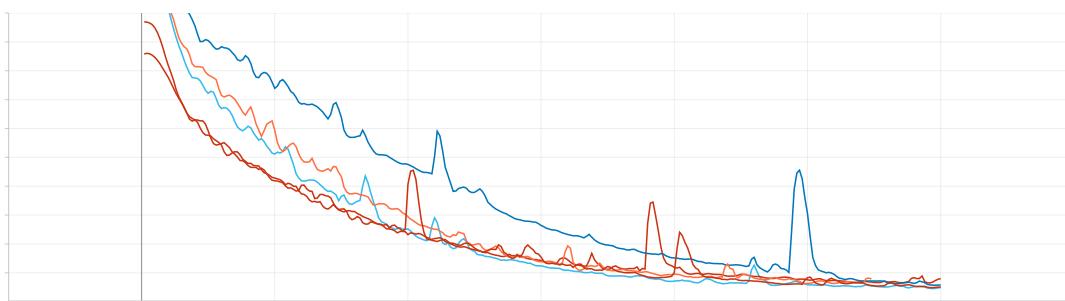
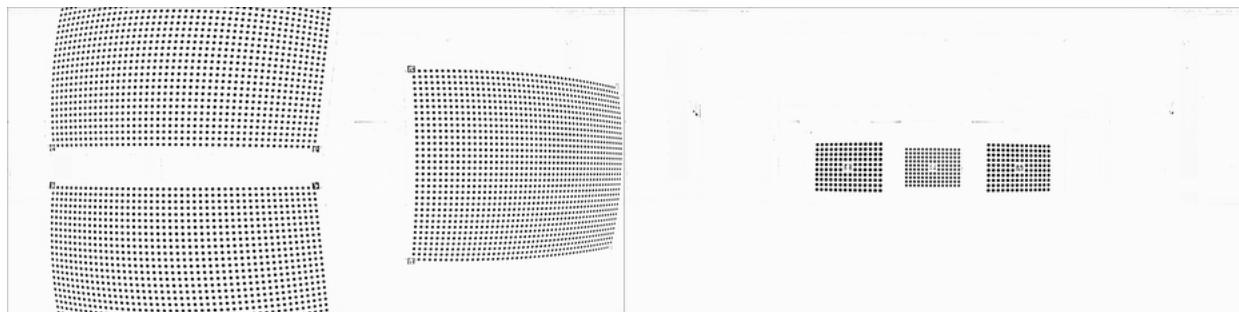


Figura 23: Validation loss $\times 5$ run al variare del random-seed da 1 a 5



Osservazioni: dai grafici e dalle immagini sopra è possibile osservare un comportamento leggermente migliore rispetto al modello precedente, evidenziando però ancora qualche difficoltà, seppur minore, con i marker Aruco.

CAE-ConvTranspose 1920×1080-pinhole + 3840×1920-fisheye configurazione di test che ha previsto l'uso di:

- CAE con ConvTranspose2d layer;
- Dataset composto sia dalle immagini 1920×1080 pinhole sia alle 3840×1920 fisheye, con resize pari a 1920×1080;
- Loss-function: *IntersectionOverUnionLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 8;

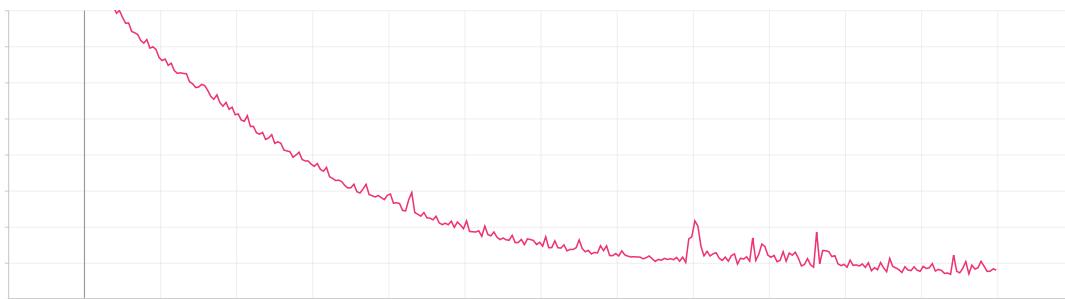


Figura 24: Training loss

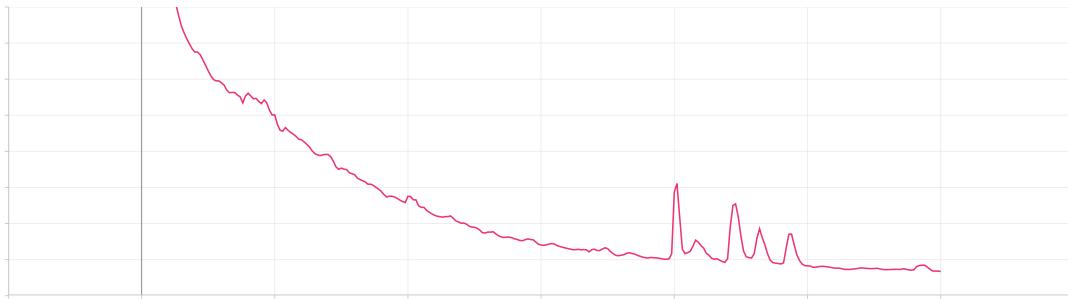


Figura 25: Validation loss

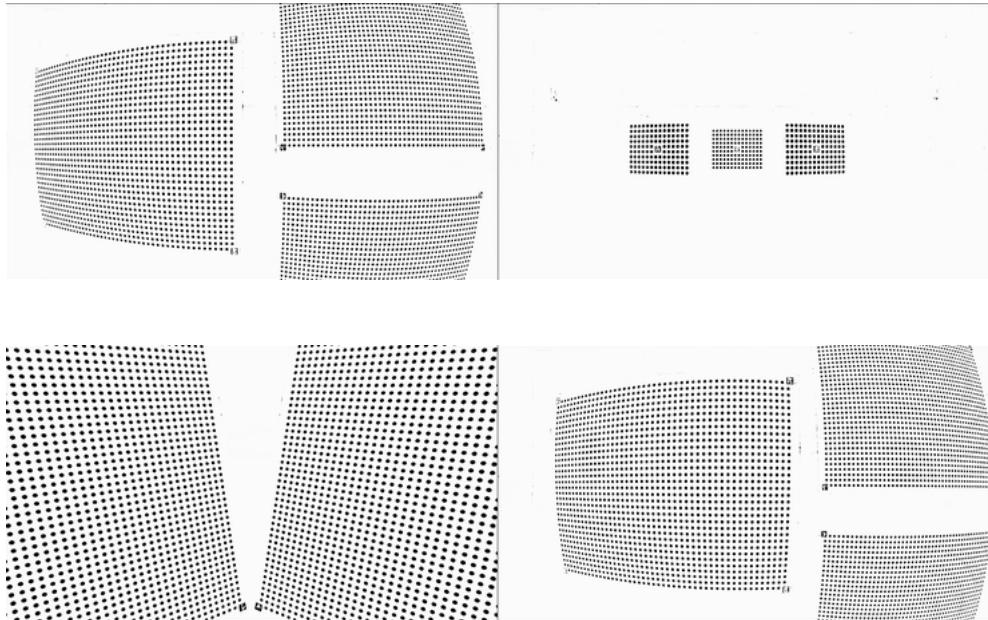


Figura 26: Risultati qualitativi estrazione marker

Osservazioni: dai grafici e dalle immagini sopra è possibile osservare una buon comportamento dal modello, nonostante la resize effettuata alle immagini fisheye. Allo stesso tempo, l'utilizzo di due formati di immagini differenti porta ad una varianza maggiore del dataset, permettendo alla rete di generalizzare in modo più efficace. Ciò nonostante, è ancora possibile evidenziare una discreta difficoltà con i marker Aruco.

CAE-Upsample, CAE-ConvTranspose 1920×1080-pinhole configurazione di test che ha previsto l'uso di:

- CAE: con Upsample layer vs. con ConvTranspose2d layer;
- Dataset composto dalle sole immagini 1920×1080 pinhole;
- Loss-function: *BinaryCrossEntropyWithLogitsLoss*;
- Normalizzazione: $mean = 0.5$, $std = 0.5$;
- Trasformazioni: abilitate;
- Batch-size: 2;

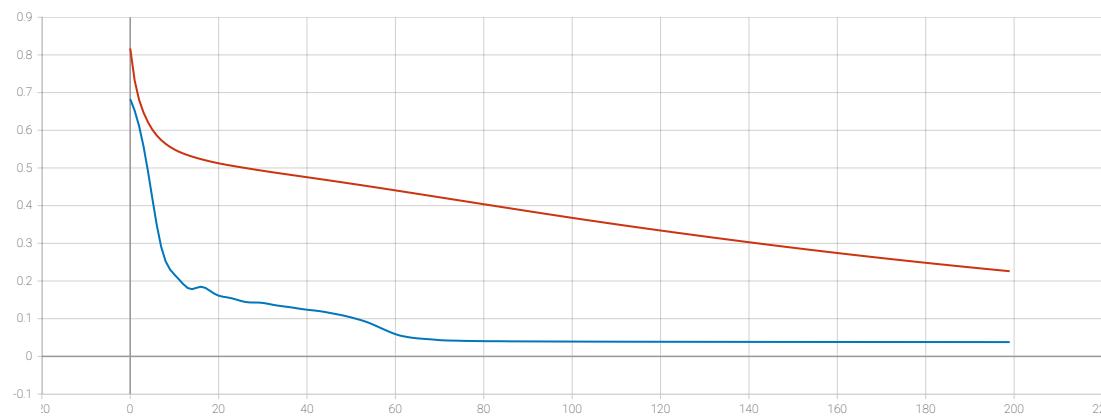


Figura 27: Training loss: Upsample (blu) vs. ConvTranspose (rosso)

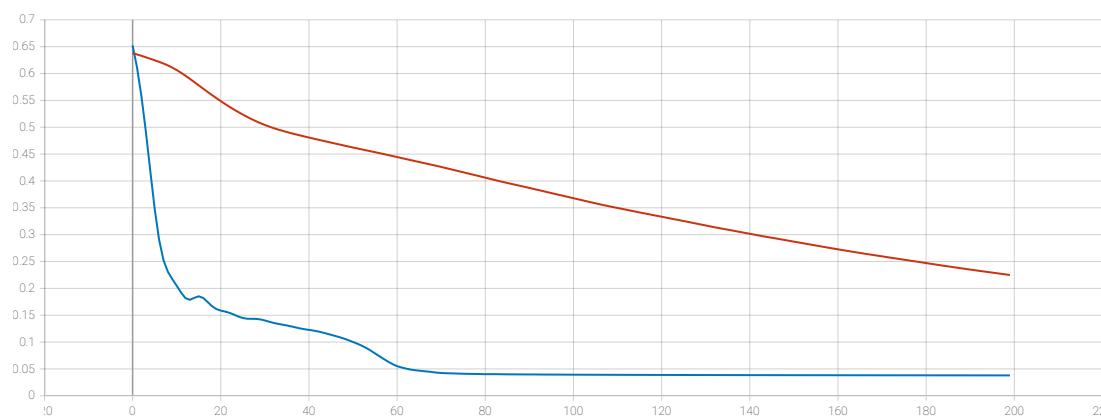


Figura 28: Validation loss: Upsample (blu) vs. ConvTranspose (rosso)

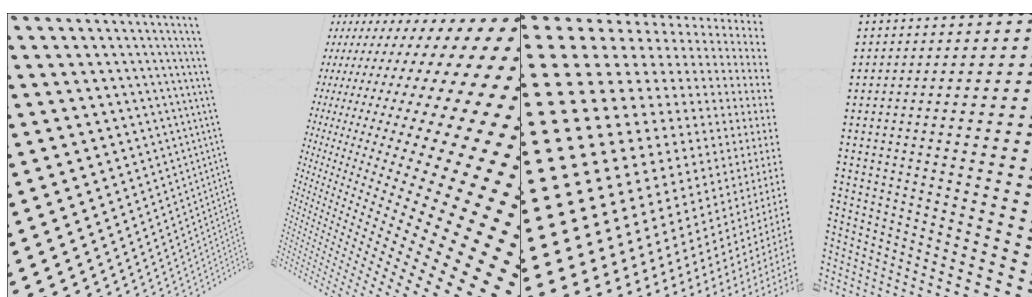
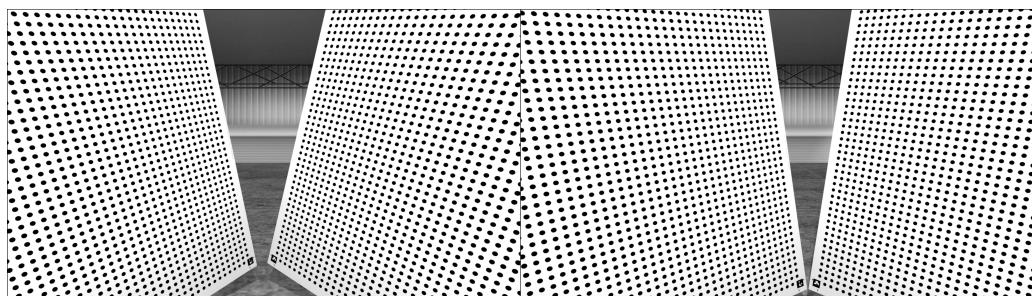


Figura 29: Risultati qualitativi estrazione marker (ConvTranspose)

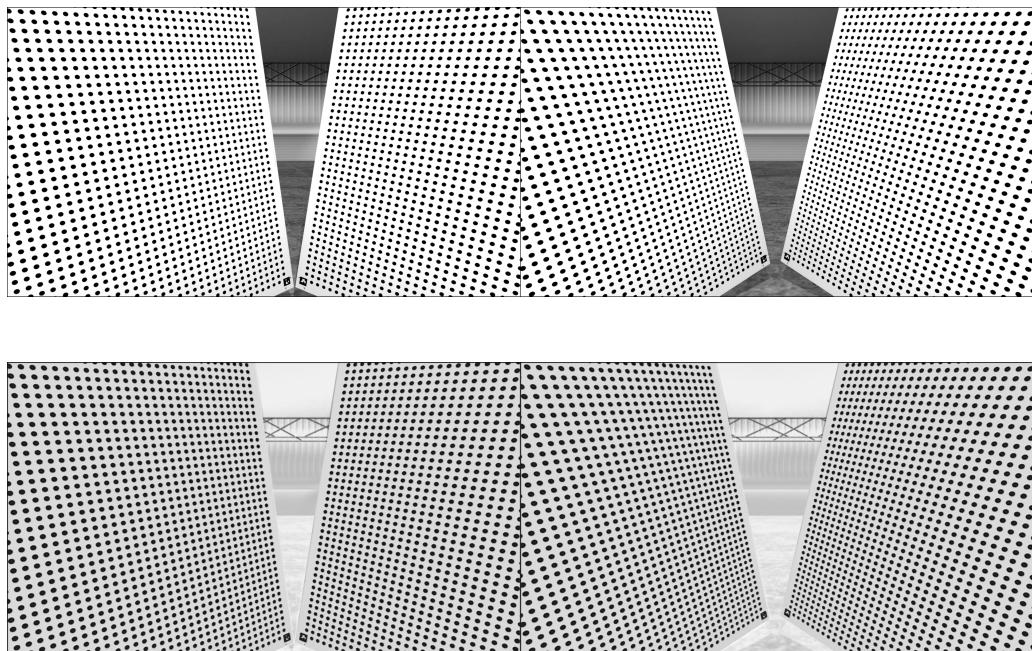


Figura 30: Risultati qualitativi estrazione marker (Upsample)

Osservazioni: dai grafici e dalle immagini qualitative sopra è possibile osservare che la *binary cross-entropy with logits loss* non risulta essere adatta al task di estrazione dei marker circolari.