Detektion und Klassifikation von Objekten aus von SONAR-Systemen erstellten Plots mithilfe von künstlicher Intelligenz

Frederik Kühne[1], Bastian Kaulen[1], Christian Kanarski[1], Finn Röhrdanz[1], Karoline Gussow[1], Gerhard Schmidt[1]

[1] *Christian-Albrechts-Universität zu Kiel, 24118 Kiel, Deutschland, Email: {frk, bk, chk, finr, kars, gus}@tf.uni-kiel.de*

Einleitung

Künstliche Intelligenzen verfügen gerade in der Bildverarbeitung mit neuen Modellen über eine herausragende Leistung besonders in der Objektklassifikation und in der Lokalisation. Diese Fortschritte versprechen in allen Bereichen der Objekterkennung sehr gute Ergebnisse. SONAR-Systeme führen die Detektion normalerweise über konventionelle Algorithmen aus, eine Klassifikation wird dabei selten durchgeführt. Dieser Beitrag wendet die durch maschinelles Lernen erstellten Bildverarbeitungsmodelle auf den SONAR-Bereich an und führt sowohl eine Detektion als auch eine Klassifikation auf simulierten SONAR-Daten durch. Die Ergebnisse versprechen eine hohe Genauigkeit bei einer minimalen Detektions- und Klassifikationszeit.

Grundlegendes Szenario

SONAR-Systeme sind je nach Konfiguration in der Lage, verschiedenste Umgebungen mit variabler Präzision zu überwachen. Während tieffrequente, leistungsstarke Projektoren Gebiete von mehreren Kilometern abdecken, können hochfrequente Systeme Strukturen im Nahfeld mit einem hohen Grad an Details abbilden. Systeme, die mittlere Frequenzen bei um die 50 kHz nutzen, bilden dabei einen guten Kompromiss zwischen Genauigkeit und Reichweite. Während die Reichweite ausreichend für Hafenumgebungen ist, können Strukturen nicht mehr detailgetreu abgebildet und deshalb nur sehr schwer klassifiziert werden. Dieses Paper beschränkt sich auf die simultane Detektion und Klassifikation von Objekten in einer ebensolchen Umgebung ausschließlich anhand der Bildgebung eines entsprechenden SONAR-Systems mithilfe einer künstlichen Intelligenz. Aufgrund der für das Training der Modelle nötigen annotierten Datenmenge wird auf eine Simulationsumgebung zurückgegriffen, die sowohl ein Hafengebiet simuliert als auch die SONAR-Bildgebung durchführt und so die geforderten Daten liefern kann.

Simulationsumgebung und Parametrisierung

Die Simulation ist flexibel parametrisierbar. Für die Trainingsdatenerzeugung wird ein reales Hafenebecken grob als ein Quader mit $l_y = 500\,\mathrm{m}$ Länge, $l_x = 250\,\mathrm{m}$ Breite und $l_z = 20\,\mathrm{m}$ Tiefe simuliert. Die Schallgeschwindigkeit wird aufgrund der geringen Tiefe als konstante $c_{\mathrm{Wasser}} = 1500\,\mathrm{m\,s^{-1}}$ mit entsprechend gradliniger, sphärischer Schallausbreitung angenommen. Die Spundwände des Hafengebietes werden mit zufälligen, positionsveränderlichen Punktzielen in einer Ebene simuliert.

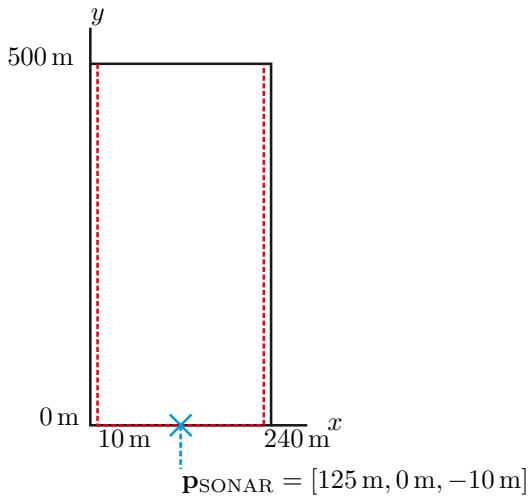

Abbildung 1: Simulationsumgebung in der x-y-Ebene mit den eingefügten Wänden in rot gestrichelt und dem SONAR-System als blaues Kreuz.

Abbildung 1 zeigt die simulierte Umgebung aus der Vogelperspektive. Die rot gestrichelten Linien zeigen die Position der simulierten Wände in U-Form, rechts und links mit einem Abstand von 10 m zur Simulationsgrenze und auf der x-Achse. Das blaue Kreuz zeigt die Position des simulierten SONAR-Systems in der Mitte des Beckens auf einer Tiefe von 10 m. Angelehnt an bestehende Hardware und für eine ausreichende Winkelauflösung wurde ein Array aus $N_{\mathrm{rx}} = 64$ Hydrophonen in Kombination mit einem Sendeelement verwendet. Die Elementabstände des Sendearrays wurden auf die verwendete Mittenfrequenz von $f_{\mathrm{c}} = 50\,\mathrm{kHz}$ abgestimmt. Für eine gute Entfernungsauflösung bei akzeptablen Beampattern wurde beim verwendeten Sendesignal eine Rauschsequenz mit einer Bandbreite von $f_{\mathrm{b}} = 20\,\mathrm{kHz}$ und einer Länge von $T_{\mathrm{Puls}} = 0.01\,\mathrm{s}$ verwendet bei einer Pingdauer von $T_{\mathrm{Ping}} = 1\,\mathrm{s}$. Das verwendete SONAR-System hat das Gebiet mit einer Abtastfrequenz von $f_{\mathrm{s}} = 192\,\mathrm{kHz}$, also mit einer Entfernungsauflösung von $\Delta r = 0.5 \cdot c_{\mathrm{Wasser}} \cdot f_{\mathrm{s}}^{-1} \approx 4\,\mathrm{mm}$ und einer Winkelauflösung von $\phi = 1°$ über 180° abgetastet. Um das Ungleichgewicht der Matrixgröße von $[N_{\mathrm{Entfernung}} \times N_{\mathrm{Winkel}}] = [192\,000 \times 181]$ auszugleichen, wurde eine Bündelung mittels Maximalwertbildung in Entfernungsrichtung auf eine Größe von $[1000 \times 181]$ durchgeführt. Die zu detektierenden und klassifizierenden Ziele sollen möglichst einfach und flexibel einstellbar sein. Dazu wurde ein Algorithmus verwendet, der beliebige 3D-Modelle möglichst realistisch in eine Wolke an

zufällig auf der Objektoberfläche verteilten Punktzielen für die Simulation umwandelt. Dafür wurde zunächst die Anzahl der gewünschten Punktziele zur Repräsentation eines Volumens auf $N_{\text{Punkte,max}} = 1000$ festgelegt. Als nächstes wurde die Größe jedes Dreiecks, aus dem ein 3D-Modell aufgebaut ist, im Verhältnis zur Gesamtgröße der Oberfläche genutzt, um die Anzahl an Punkten festzulegen, die jedes einzelne Dreieck repräsentieren sollen. Anschließend wurden diese Punkte zufällig auf der Dreieckoberfläche angeordnet, um so das Objekt komplett darzustellen. Ein zufällig generierter Winkel und Position platzieren die Punktwolke dann an eine Stelle in der simulierten Hafenumgebung. Um die Simulation möglichst realitätsgetreu zu gestalten, wurden zusätzlich alle Punkte entfernt, die im Schatten von davor liegenden Punkten liegen.

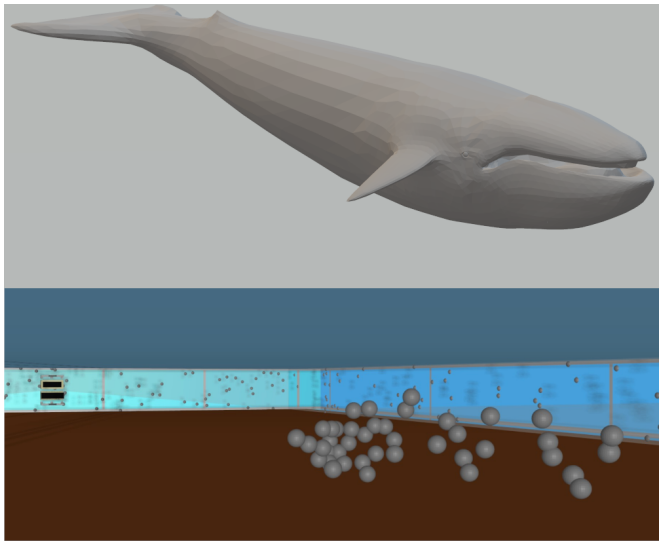

Abbildung 2: Ein Wal als 3D-Modell und als Punktwolke in der Simulation.

Abbildung 2 zeigt eine der drei verwendeten 3D-Modelle und die entstandene Punktwolke in der simulierten Hafenumgebung. Insgesamt wurden drei verschiedene Objekte verwendet. Neben dem in Abbildung 2 verwendeten generischen Wal wurde ein U-Boot und ein Schiff in Form einer Punktwolke simuliert. Die Simulation erzeugt für jeden Durchgang ein anderes Szenario mit zufälligen Parametern. Dabei wird zunächst die Anzahl der in der Simulation vorkommenden Objekte auf eins bis drei festgelegt. Für jedes Objekt wird dann wie beschrieben ein Ort und eine Ausrichtung erzeugt. Nach jeder Simulation erzeugt das SONAR-System eine entsprechende SONAR-Matrix, die zusammen mit den einzelnen Punktzielpositionen abgespeichert werden.

Zugrundeliegendes Modell

Klassische neuronale Netzwerke sind in der Lage, Klassifikaitionsaufgaben auf gegebenen Bildern jeglicher Art je nach Training und Modell durchzuführen. Das erfordert allerdings einen vorgeschalteten Detektionsalgorithmus, um die geforderten Bildausschnitte des Ziels aus der gesamten SONAR-Matrix zu extrahieren. Das *Mask Region Convolutional Neural Network* (Mask R-CNN)[1] ist in der Lage, die Detektionsaufgabe neben der Klassifikation mit in ein Modell zur Instanzsegmentierung zu integrieren. Es ist eine Zusammenschaltung von mehrern Netzen.

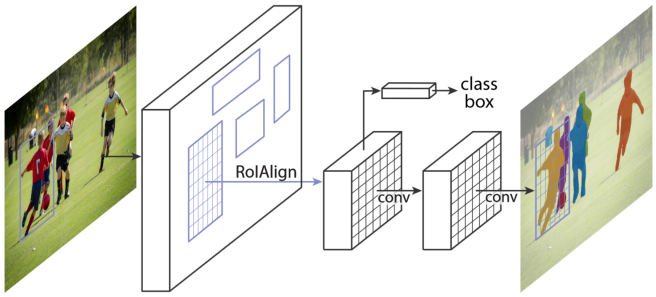

Abbildung 3: Das *Mask Region Convolutional Neural Network*[1].

Abbildung 3 zeigt die grobe Struktur des Netzes. Zum Anfang wird ein *Region Proposal Network* (RPN) geschaltet, das dafür verantwortlich ist, Bildausschnitte vorzuschlagen, in denen sich Objekte befinden können. Es übernimmt also die Detektionsfunktion des Netzwerkes. Die vorgeschlagenen Regionen werden dann durch *RoIAlign* so interpoliert, dass sie die selbe Größe haben, wie die Kernel des nachfolgenden Klassifikationsnetzes. Die Interpolation sorgt für eine deutlich detailreichere Informationsübertragung gegenüber anderen Modellen, die beispielsweise nur ein Pooling verwenden. Für das nachgeschaltete tiefe neuronale Netz wurde in diesem Paper das *ResNet50*[2] als Klassifikator verwendet. Parallel dazu wird eine pixelgenaue Segmentierung des erkannten Objektes im Bildausschnitt durchgeführt, die die Detektion von Objekten in der SONAR-Matrix deutlich genauer ermöglicht.

Erzeugung der Trainingsdatenbank

Mit der vorgestellten Simulation ist es möglich, in einer vollständig kontrollierten Simulationsumgebung annotierte Trainingsdaten zu erzeugen. Als Vorbild diente dabei der von Microsoft veröffentlichte Datensatz *Common Objects in Context* (COCO)[3]. In 382 000 Bildern werden 250 000 Instanzen von 91 Klassen abgebildet und annotiert. Diese Datenbank dient zum einen als Benchmark für ein trainiertes Modell und zum andern bringt es ein Standard-Datenformat mit, in dem annotierte Bilder einfach gespeichert und in den meisten Entwicklungsumgebungen leicht geladen werden können. Für ein gutes Modell wird eine Klasse also im Schnitt etwa 30 000 mal in den Trainingsdaten abgebildet. Die beschriebene Simulation erzeugt SONAR-Matrizen mit im Durchschnitt zwei Objekten pro Periode, für eine vergleichbare Anzahl an Daten pro Klasse sind also insgesamt etwa 45 000 Perioden nötig. Diese Grenze wurde mit 100 000 erzeugten annotierten Bildern erreicht.

Für das Training wurden 70 % der Daten genutzt, während für den Test die restlichen 30 % verwendet wurden. Ein Validierungsset wurde nicht verwendet, da kein Hyperparametertuning durchgeführt wurde. Tabelle 1 zeigt die Anzahl der verwendeten Instanzen aufgeteilt nach Klasse und Datensatz. Um die erzeugten Daten in das

Tabelle 1: Anzahl der Instanzen in den erzeugten Daten aufgeteilt in Test-Set und in Trainings-set.

	U-Boote	Schiffe	Wale
Training	46 708	46 895	46 460
Test	20 074	20 013	19 999
Gesamt	66 782	66 908	66 459

COCO-Datenformat umzuwandeln, wurden alle Punkte der simulierten Volumina auf die 2D-Ebene projiziert, in der auch das Array die SONAR-Matrix erzeugt. Als nächstes wird die konvexe Hülle um Punktwolken einer Klasse gebildet, um die pixelgenaue Position der Objekte zu berechnen. Aus diesen Positionen kann ebenfalls das für das Training des RPN nötige Begrenzungsrechteck mit Minimalwert- bzw. Maximalwertbildung extrahiert werden. Die so präparierten Daten können ähnlich einer relationalen Datenbank in einer json-Datei im COCO-Datenformat gespeichert werden. Damit wurde eine Pipeline geschaffen, die nahezu beliebige Szenarien simulieren, analysieren und in einem geeigneten Format zum Training abspeichern kann.

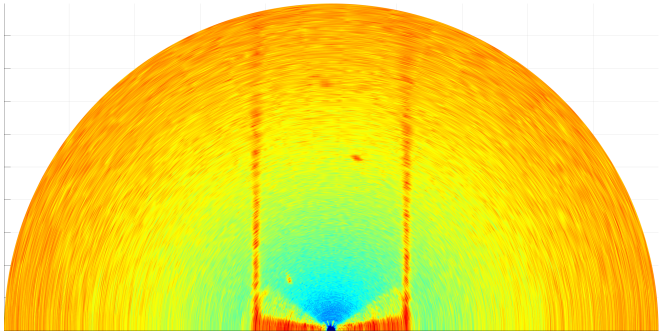

Abbildung 4: Ausgabe der SONAR-Matrix nach einer Simulationsperiode.

Abbildung 4 zeigt eine SONAR-Matrix nach einer zufälligen Simulationsperiode. In dieser Periode wurde ein Wal relativ nahe zum SONAR-System, ein weiterer Wal etwa in der Mitte des Beckens und ein U-Boot etwas weiter weg simuliert. Auffällig ist der sichtbare Abfall des Signal-zu-Rausch-Verhältnisses (SNR) mit steigender Entfernung. Das resultiert aus der automatischen Kompensation der geometrischen Dämpfung durch das SONAR-System. Auch ist kein deutlicher Unterschied zwischen den Objektklassen erkennbar. Für das Netz ist als Eingang eine Graustufenversion des Bildes als 2D-Matrix ohne Informationsverlust ausreichend.

Ergebnisse des Modells

Das Netz wurde auf einem Computer mit einer AMD Radeon RX 6800 Grafikkarte mit 16 GB VRAM sowie einem AMD Ryzen 5 5600X 6-Core-Prozessor mit 16 GB RAM trainiert. Dafür wurde die Entwicklungsumgebung Detectron2[4] genutzt. Das Training wurde über 30 000 Iterationen in 10 h durchgeführt.

Abbildung 5 zeigt die Klassifizierungsgenauigkeit über die Trainingsiterationen an. Es ist zu erkennen, dass nach

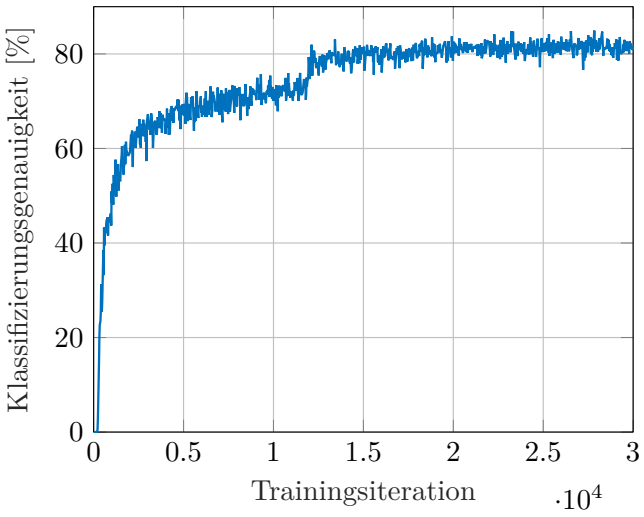

Abbildung 5: Die Genauigkeit der Objektklassifizierung der Trainingsdaten über die Trainingsiterationen.

etwa 12 000 Iterationen ein Sprung von etwa 70 % auf 80 % erfolgt, die sich ab dann aber nicht wesentlich verbessert.

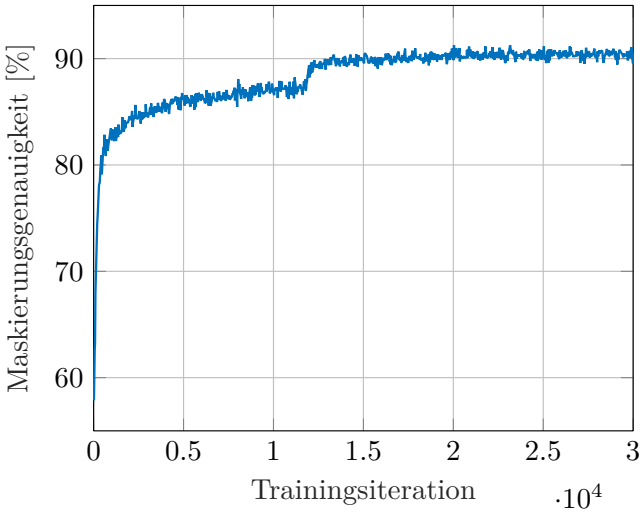

Abbildung 6: Die Genauigkeit der Objektsegmentierung der Trainingsdaten über die Trainingsiterationen.

Abbildung 6 zeigt die Genauigkeit der pixelgenauen Segmentierung der Objekte mit einem ähnlichen Sprung bei etwa 12 000 Traningsiterationen. Die Segmentierungsgenauigkeit kann im SONAR-Szenario verglichen werden mit der Detektion von Objekten in der SONAR-Matrix. Die Genauigkeit erreicht einen guten Wert von etwa 90 % am Ende des Trainings. Aussagekräftigere Ergebnisse bringt die Anwendung des Netzes auf den im Training nicht verwendeten Testdatensatz. Hierbei kann zunächst die Klassifikationsgenauigkeit anhand einer Verwechslungsmatrix geprüft werden.

Abbildung 7 zeigt die Verwechslungsmatrix erweitert um die Fälle, in denen der Detektionsteil etwas gefunden hat, was kein Objekt war oder nichts gefunden hat, wo eigentlich ein Objekt war. Für die folgende Klassifikationsauswertung wurden allerdings nur korrekt detektierte Ob-

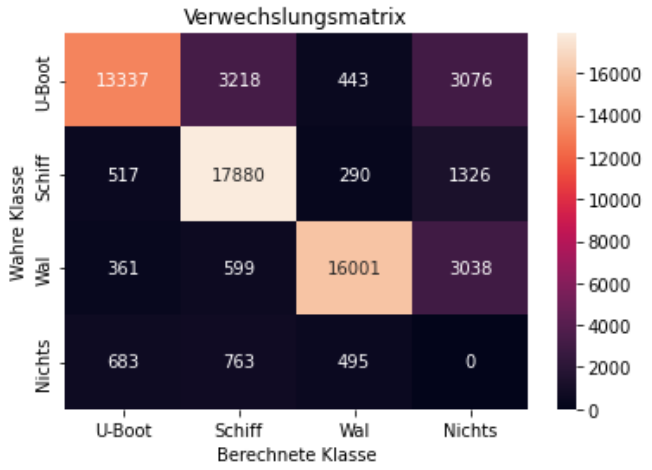

Abbildung 7: Erweiterte Verwechslungsmatrix zur Analyse der Klassifikationsgenauigkeit des Modells.

jekte genutzt. Es werden die für Klassifikationen üblichen Parameter *Precision*, *Recall* und *Accuracy* ausgewertet.

Tabelle 2: *Precision*, *Recall* und *Accuracy* des Modells für die Testdaten aufgeteilt in Objektklassen.

	U-Boote	Schiffe	Wale
Precision	93.82 %	82.41 %	95.62 %
Recall	78.46 %	95.69 %	94.34 %
Accuracy	91.38 %	91.21 %	96.79 %

Tabelle 2 zeigt die Auswertung der Klassifikation. Mit einer Genauigkeit von über 90 % für jede Klasse liefert das Modell eine gute Klassifikation trotz der visuell schlecht unterscheidbaren Klassen. Auffällig ist, dass das U-Boot oft mit dem Schiff verwechselt wurde, was wahrscheinlich an der ähnlichen Struktur und Länge liegt, die nicht so gut auseinandergehalten werden kann. Auch werden einige Objekte nicht gefunden, wohingegen es vergleichsweise wenig gefundene Objekte gibt, wo eigentlich keine sind. Für die Detektionsgenauigkeit kann die Präzisionsauswertung der Begrenzungsrechtecke und der pixelgenauen Segmentierung herangezogen werden. Dabei wird die *Intersection over Union* (IoU) genutzt, um den Anteil der Detektionen zu ermitteln, die verschiedene Kriterien in der Überlappung der detektierten und wahren Fläche erfüllen. Eine mittlere Präzisionsgenauigkeit (AP) kann dabei aus dem Durchschnitt verschiedener spezifischer Präzisionsgenauigkeiten berechnet werden. Der AP50 und AP75 setzen jeweils eine Überlappung von 50 % bzw. 75 % voraus, während APs, APm und APl die Objektgröße in Betracht ziehen, mit kleinen, mittleren und großen Objekten im Verhältnis zum Gesamtbild.

Tabelle 3 zeigt die Ergebnisse der Detektion von Objekten durch die künstliche Intelligenz. Die durchschnittliche Präzision des Begrenzungsrechtecks ist mit 51.33 % etwas besser als die des Vergleichsdatensatzes COCO mit 41 %. Die pixelgenaue Detektion erreicht eine etwas schlechtere Präzision von 31.96 % im Vergleich zum COCO-Datensatz mit 37.2 %. Dies liegt vermutlich an der sehr

Tabelle 3: Präzisionswerte der Begrenzungsrechtecke sowie der pixelgenauen Segmentierung für verschiedene Kriterien.

	Rechteck	Pixelgenau
AP50	76.51 %	62.63 %
AP75	60.65 %	31.01 %
APs	51.3 %	31.92 %
APm	82.57 %	78.37 %
APl	-	-
AP	51.33 %	31.96 %

kleinen Größe der Objekte im Verhältnis zum Gesamtbild und dem dementsprechend hohen anteiligen Fehler bei Abweichungen von nur wenigen Pixeln.

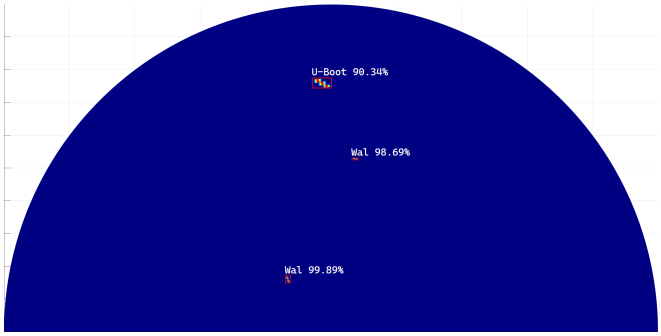

Abbildung 8: Erweiterte Verwechslungsmatrix zur Analyse der Klassifikationsgenauigkeit des Modells.

Abbildung 8 zeigt die SONAR-Matrix aus Abbildung 4 nach Anwendung der künstlichen Intelligenz auf die Daten. Alle Ziele werden erkannt und korrekt klassifiziert, der Rauschteppich kann komplett entfernt werden.

Zusammenfassung und Ausblick

Mit dem vorgestellten System lässt sich ein beliebig komplexes Modell für verschiedenste Hafenumgebungen trainieren, um eine sehr gute Klassifikation und Detektion in kürzester Zeit zu ermöglichen. Allerdings funktioniert dieses Modell nur so gut, wie die Simulation die Wahrheit abbilden kann. Für eine Anwendung auf reale SONAR-Systeme sind also idealerweise echte Daten nötig. Auch sollten aufeinanderfolgende Pings berücksichtigt werden, um physikalische Eigenschaften von maritimen Objekten mitmodellieren zu können.

Literatur

[1] K. He, G. Gkioxari, P. Dollár und R. Girshick, Mask R-CNN, 2018

[2] K. He. X. Zhang, S Ren und Jian Sun, Deep Residual Learning for Image Recognition, 2015

[3] T. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. Zitnick und P. Dollár, Microsoft COCO: Common Objects in Context, 2015

[4] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo und R. Girshick, Detectron2, 2019