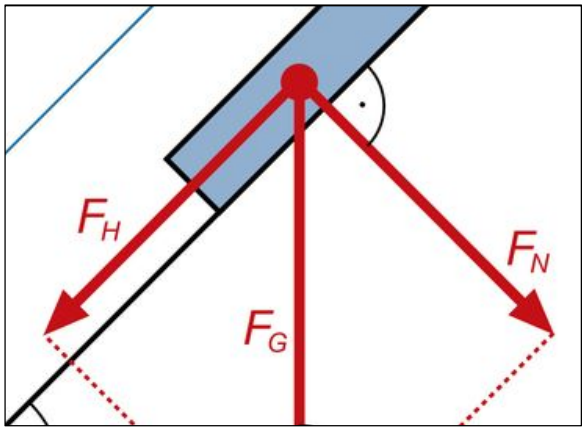
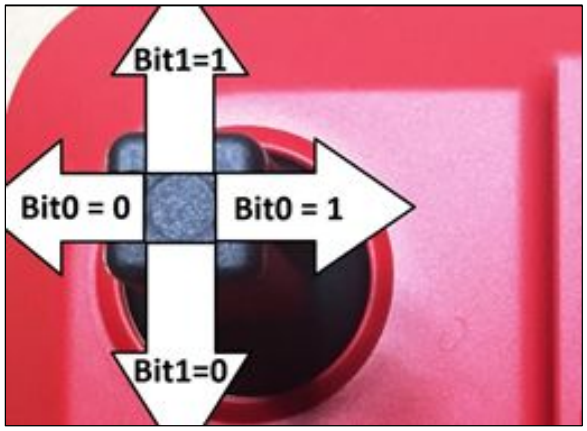
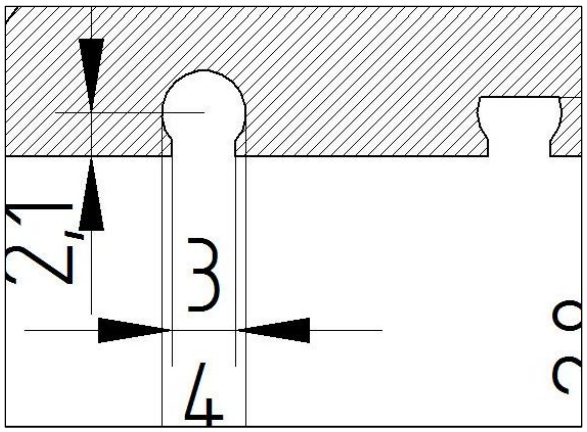
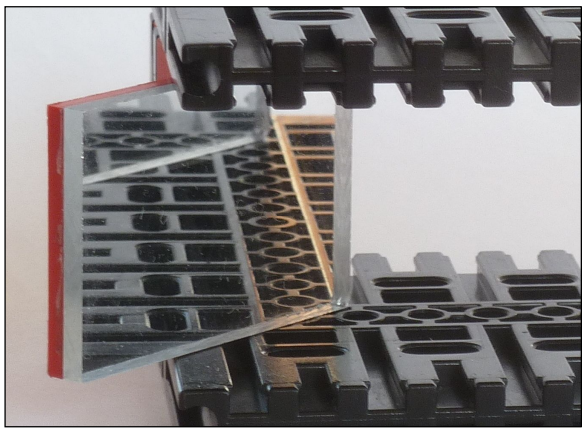
ft:pedia

Heft 3/2016









Herausgegeben von
Dirk Fox und Stefan Falk

ISSN 2192-5879

Editorial

Was wir schon 1972 wussten.

Dirk Fox, Stefan Falk

Dass das fischertechnik-System ein pädagogisch herausragendes Baukastensystem ist, muss man einem fischertechniker nicht erklären. Dass sich diese Erkenntnis wissenschaftlich untermauern lässt, wird ein fischertechniker vermuten. Dass eine solche gesicherte Analyse aber bereits vor fast 44 Jahren vorlag, mag dann doch selbst den einen oder anderen fischertechniker freudig überraschen.

Im Jahr 1972 (wir erinnern uns: die 20. Olympischen Spiele in München begeistern die Welt, die Watergate-Affäre erschüttert die USA, Andreas Baader und Ulrike Meinhoff werden verhaftet, HP bringt den ersten wissenschaftlichen Taschenrechner HP-35 auf den Markt, der Club of Rome veröffentlicht „Die Grenzen des Wachstums“, die heutige SAP AG wird gegründet, Heinrich Böll erhält den Literaturnobelpreis) erscheint in der Dezemberausgabe des „Test“-Magazins der Stiftung Warentest ein Bericht über zehn aktuelle Spielzeug-Baukastensysteme.

Professor Hans Mieskes aus Gießen in seiner Einleitung: „*Im Hantieren mit Bauelementen erfährt das Kind die dritte Dimension; es wird ihrer nicht nur inne, sondern gestaltet, bewältigt sie auch, erhält dadurch das Formgefühl, lernt die Eigenschaften des Materials und die Bedingungen seiner Verwendung kennen. Ein nach den Naturgesetzen orientiertes Denken, Erkennen, Planen, Kombinieren und Konstruieren kommt in Gang. (...) Der Bau steht oder fällt zusammen. Mogeln ist nicht möglich. Hierin liegt nicht nur der bildende Ertrag aller Bauspiele (von dem Ertrag praktischer Fertigkeiten ganz zu schweigen); es steckt auch eine eminent erzieherische Komponente darin.*“ (in: Test, 7. Jhrg., Ausgabe 12/1972, S. 505).

Im einleitenden Artikel halten sich die Tester hinsichtlich einer Empfehlung zunächst bedeckt – und raten dazu, auszuprobieren, welches System am besten zum jeweiligen Kind passt. Deutlicher werden sie jedoch in der Einzelbewertung der fischertechnik-Baukästen:

„*Das z. Zt. am besten durchdachte und leistungsstärkste Baukastensystem mit einem hohen Aufforderungscharakter für alle Benutzer. (...) Bezeichnenderweise bekunden Eltern und Sozialpädagogen immer noch (unberechtigte) Bedenken gegenüber dem technischen Charakter des Systems, nicht aber die Kinder. Die erforderlichen Fingerfertigkeiten werden auch von Kleinen schnell und leicht erlernt; von da an kommen Phantasie, Beobachten, Denken, Planen, Konstruieren, Experimentieren und Erkennen zu ihrem vollen Einsatz. Alle Bauformen und Funktionsweisen sind und bleiben durchschaubar; sie vermitteln verlässliches technisches Wissen und Können.*“

Ein Schmunzeln zaubert schließlich die Altersempfehlung auf die Lippen eines fischertechnikers: „*Ab vier Jahren (...), nach oben hin unbegrenzt.*“

Na dann also – ran an die Kästen.

Beste Grüße, Euer ft:pedia-Team

P.S.: Am einfachsten erreicht ihr uns unter ftpedia@ftcommunity.de oder über die Rubrik *ft:pedia* im Forum der ft-Community.

Inhalt

Was wir schon 1972 wussten. 2
Mini-Modelle (Teil 15): Nudelholz 4
Wiedergefunden 5
Luftseilbahn – frei schwebend über die Berge 8
Das Hyper-Pseudoskop 13
Verstellbare Propeller 16
Der etwas andere Motor 22
Wecke den Erfinder in Dir 26
fischertechnik-Nutprofile selbst herstellen (2) 29
Tips on using the fischertechnik TXT Controller 31
Große Modelle mit nur einer fischertechnik-IR-Fernsteuerung ansteuern 33
Codes der fischertechnik-Infrarot-Fernsteuerungen 35
RoboRISC: Visual Basic für den Robotics TXT 43
Hochregallager mit Kamera-Strichcodeleser, Microsoft Visual Basic 2010 und RoboRISC 48

Termine

Was?	Wann?	Wo?
fischertechnik Convention 2016	24.09.2016	Dreieich
Modellschau Münster	20.11.2016	Kardinal-von-Galen-Gymnasium (Hiltrup)

Impressum

http://www.ftcommunity.de/ftpedia

Herausgeber: Dirk Fox, Ettlinger Straße 12-14, 76137 Karlsruhe und Stefan Falk, Siemensstraße 20, 76275 Ettlingen

Autoren: Daniel Canonica, Andreas Gail, Ralf Geerken, Peter King, Peter Krijnen, Thomas Püttmann, Rüdiger Riedel, Harald Steinhaus, Andreas Tacke, René Trapp, Dirk Uffmann, Dirk Wölffel.

Modell

Mini-Modelle (Teil 15): Nudelholz

René Trapp

„... allen #ftctalk-Teilnehmern gewidmet ...“

Die Bezeichnung „Mini-Modell“ erscheint unverschämt, angesichts der Stückliste:

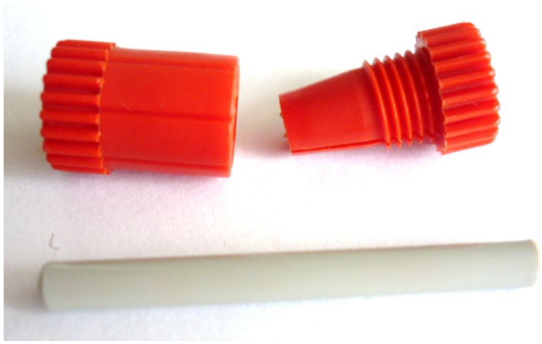


Abb. 1: Die Einzelteile für das Nudelholz

Abb. 2: Das fertige Nudelholz

Diese Teile stecken und schrauben wir so zusammen, dass sie mittig auf der grauen Achse sitzen, und fertig ist das Nudelholz.

Danke an Hanni für die Präsentation (Abb. 2). Odd Job* demonstriert uns noch eine weitere geläufige Anwendungsmöglichkeit (Abb. 3).

Abb. 3: Odd Job’s Anwendung

Hier die Stückliste mit Teilenummern:

St.	ft-Nr.	Bezeichnung
1	38414	K-Achse 40 mm
1	35113	Spannzange
1	31915	Zangenmutter

* Name von der Redaktion geändert.

Modell

Wiedergefunden

Peter Krijnen

Es gibt fischertechnik-Modelle, die gehen einem seit der Kindheit nicht aus dem Kopf. Und manch eines davon muss man eines Tages einer Weiterentwicklung unterziehen…

Ende 1967 haben mein Bruder und ich unseren ersten fischertechnik Kasten bekommen. Weil mein Bruder zwei Jahre älter ist als ich, erhielt er den Grundkasten 100 und ich nur den Grundkasten 50. Das war aber kein Problem, denn an Geburtstagen, Ferien und Festlichkeiten haben wir immer kleine Kästen bekommen. Was bedeutete, dass wir immer größere Modellen bauen konnten.

Am Anfang habe ich mehrfach die Modelle aus den Anleitungen gebaut. Aber speziell das Modell oben auf Seite sechs hat mir besonders gefallen: ein Räumgerät (Abb. 1). Nach einem Umzug war dann genau diese Anleitung verschwunden. Erst vor wenigen Wochen habe ich sie im Internet wiedergefunden (Ref.-Nr. ft 10-400/12/67).

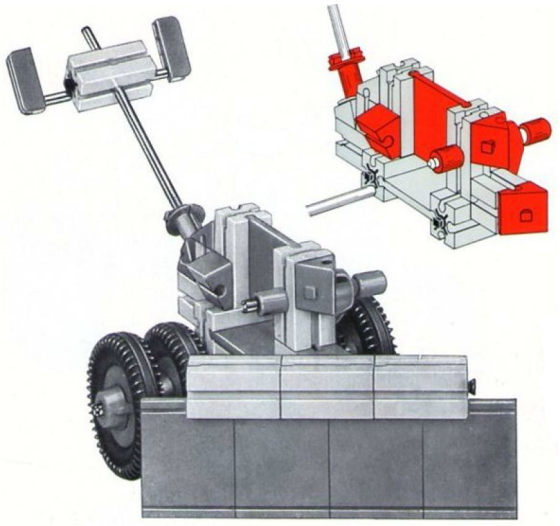


Abb. 1: Modell „Räumgerät“ aus der Anleitung zum Grundbaukasten 100 (1967)

Natürlich habe ich das Räumgerät gleich nachgebaut. Danach habe ich mir die Bilder in der Anleitung einmal genauer angesehen: Da stimmte etwas nicht. Die Zeichnung sieht anders aus als das Foto.

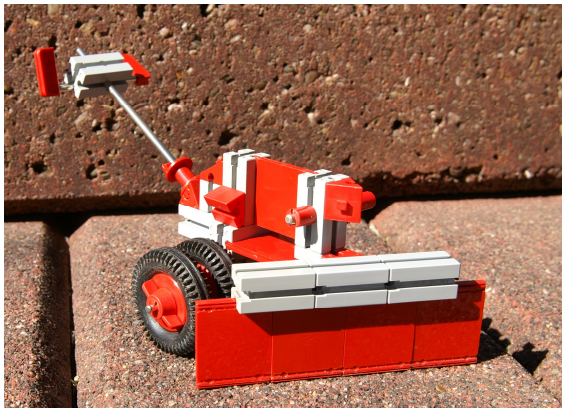


Abb. 2: Das Räumgerät nach dem Foto

In der Zeichnung ist das Planierschild mit einem Baustein 15 (mit zwei Zapfen) an einen BS30 gebaut. Auf dem Foto ist zu sehen, dass beide Steine hier verschwunden sind, und dass das Planierschild jetzt mit einem BS15 angebaut ist.

Abb. 3: Seitenansicht des Räumgeräts

Auf dem Foto ist die Achse für die Räder (vermutlich) in zwei BS15 gelagert, und auf der Zeichnung in einem BS30. Ich habe also einfach beide Versionen gebaut.

Abb. 4: Das Räumgerät nach der Zeichnung

Dabei hab ich feststellen müssen, dass ich, nach all den Jahren, eine komplett andere Vorstellung von diesem Modell bekommen habe. Im Laufe der Jahren habe ich mehrfach solche Geräte beobachten können: Die hatten alle viel mehr „Körper“. Wie würde dieses Modell also heute aussehen? Natürlich mit einem richtigen Planierschild.

Als erstes habe ich mir die Anleitung des Kastens *Tractors* (520397) angesehen und mich dann für die Konstruktion des kleinen Traktors entschieden (Abb. 5).

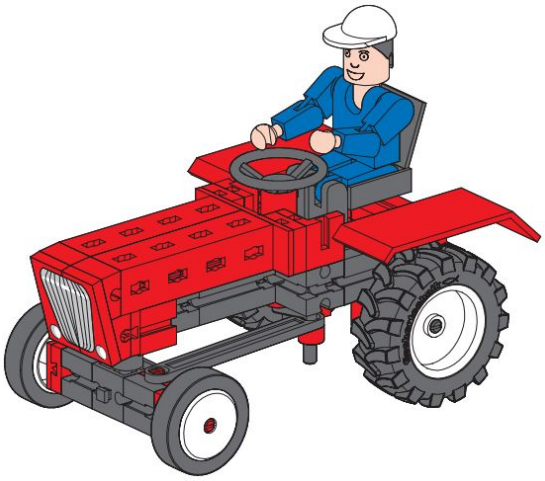


Abb. 5: Kleiner Traktor aus Tractors

Das heißt, ich habe dessen Motorhaube als Aufbau benutzt. Der Aufbau des Chassis stand für mich schon fest: Mit einem Planierschild 150 rot und zwei Traktorreifen. Statt der beiden Metallachsen und einem BS30 für das Bedienungspanel habe ich drei Rastachsen verwendet, einen BS7,5, einen Winkelstein 60, eine Radachse und eine Bauplatte 15x15 (Abb. 6-8).

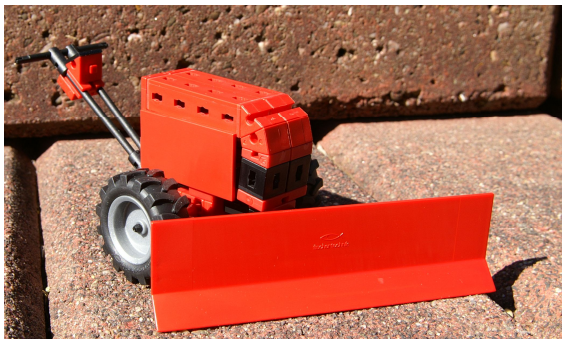


Abb. 6: Modernisiertes Räumgerät

Abb. 7: Seitenansicht des Räumgeräts

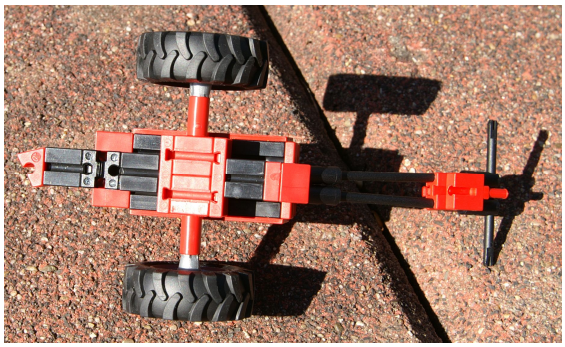


Abb. 8: Ansicht von unten

Als nächstes habe ich mir die Anleitung des Kastens *Power Tractors* (91079) angesehen und mich dort für den kleinen Traktor auf Seite 53 entschieden (Abb. 9).

Auch hier habe ich die Motorhaube benutzt, aber jetzt in gelb. Da ich für dieses Modell Streben benutzen wollte, habe ich das Chassis etwas ändern müssen: mit dem Planierschild 120 gelb und zwei LKW-Reifen. Das Bedienungspanel habe ich diesmal mit zwei Streben angebaut.

Abb. 9: Traktor-Modell aus Power Tractors

Der BS7,5, der Winkelstein 60, eine Radachse und die Bauplatte 15x15 sind geblieben (Abb. 10-12).

Abb. 10: Erneut modernisiertes Räumgerät

Abb. 11: Seitenansicht des Räumgeräts

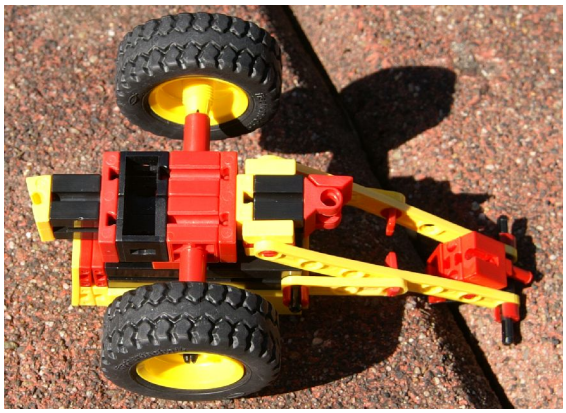


Abb. 12: Ansicht von unten

Statt eines Planierschilds könnte man auch andere Geräte anbauen oder die Räder durch Bodenfräsen (Zahnräder) ersetzen.

In der Gegend, in der ich wohne, gibt es viele Baumschulen. Die benutzen dieses multifunktional einsetzbare Motorgerät auch mit einem Anhänger. Als Vorbild wählte ich den Anhänger aus Abb. 9.

Abb. 13: Motorgerät mit Anhänger

Die Grundplatte 120 ist die Basis; beim Aufbau ersetzte ich die Streben durch gelbe Bauplatten. Die Verbindung mit dem Motorgerät erfolgt mit drei Streben, die mit zwei Strebenadaptern unter dem Anhänger montiert sind. Mit einem Gelenkwürfel und einer Adapterlasche wird der Anhänger an das Motorgerät gekuppelt (Abb. 13-15).

Abb. 14: Seitenansicht des Motorgeräts

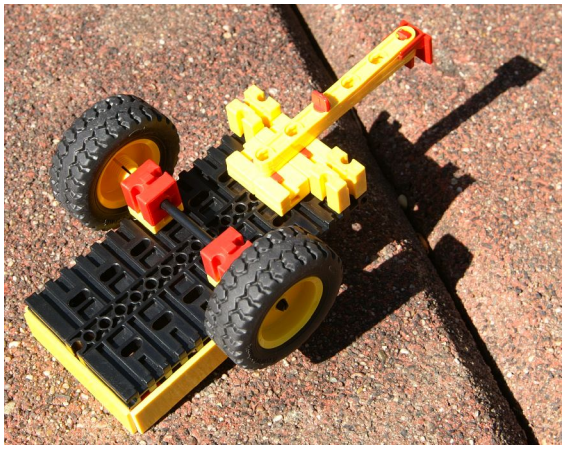


Abb. 15: Anhänger von unten

Viel Spaß beim Nachbauen!

Modell

Luftseilbahn – frei schwebend über die Berge

Daniel Canonica

Über steile Felsabhänge, wo keine Schienen gebaut werden können, schweben anmutig Luftseilbahnen. Ein Kindertraum, verwirklicht mit stabilen Elementen.

Manches Kind träumt von einem Modell einer Luftseilbahn, mit Legoteilen oder einem Metallbaukasten verwirklicht. Und so mancher Traum zerplatzt, weil die Seilspannung zu groß wird oder die Kabine unversehens talwärts rast.

Nach gefühlten 40 Jahren durfte ich meinen Traum in die Wirklichkeit umsetzen.

Technische Meisterwerke

Luftseilbahnen können große Höhenunterschiede rasch überwinden. In der Regel hängen sie an einem oder zwei Tragseilen und werden von einem Zugseil gezogen. Wie bei den Standseilbahnen hängen zwei Kabinen am selben Zugseil, welches an der Berg- und Talstation umläuft, sodass der Antriebsmotor vor allem den Gewichtsunterschied zwischen der bergwärts und der talwärts fahrenden Kabine überwinden muss (in der Praxis spielt auch das Seilgewicht eine Rolle.) Dieser Typ wird auch Pendelseilbahn genannt [1].

Außerdem gibt es vor allem in Skiorten auch Gondelbahnen, welche sich in ein ständig laufendes Zugseil ein- und auskuppeln und in den Stationen langsam umlaufen. Der Vorteil der Gondelbahnen liegt vor allem in der hohen Beförderungskapazität.

Die Kabinen vieler Luftseilbahnen fassen heute 100 Personen und mehr und wiegen 10-15 Tonnen; sie bewegen sich mit etwa 20-40 km/h berg- und talwärts.

Abb. 1: Luftseilbahn in Zell am See, Österreich (Quelle: Wikipedia)

Auch wenn einem beim Herunterschauen manchmal fast schwindlig wird, so ist die durchschnittliche Steigung vieler Luftseilbahnen nicht extrem hoch: die meisten Bahnen überwinden bei mehreren Kilometern horizontaler Länge einen Höhenunterschied von einigen hundert bis zu etwa 2.000 Metern. So hat etwa die Bahn zum *Klein Matterhorn* mit der höchstgelegenen Bergstation in Europa (3.820 m ü. M.) eine durchschnittliche Steigung von ca. 25%. Allerdings haben viele Bahnen flachere und zwischendurch auch sehr steile Abschnitte.

Eine Bahn, die offenbar viele Rekorde bricht, ist die *Dagu Glacier*-Bahn in Sichuan, China: Sie überwindet auf 2,4 km mehr als 1200 m Höhendifferenz und die Bergstation liegt auf 4.843 m über Meereshöhe [2].

Das Modell

Von Anfang an war klar, dass mit dem Modell verschiedene Herausforderungen zu bewältigen sind:

- Insbesondere die Bergstation muss eine erhebliche Spannung des Tragseils aushalten.
- Die Kabinen müssen so leicht sein, dass sie das Tragseil nicht übermäßig belasten.
- Das Zugseil muss ebenfalls eine relative hohe Spannung haben.

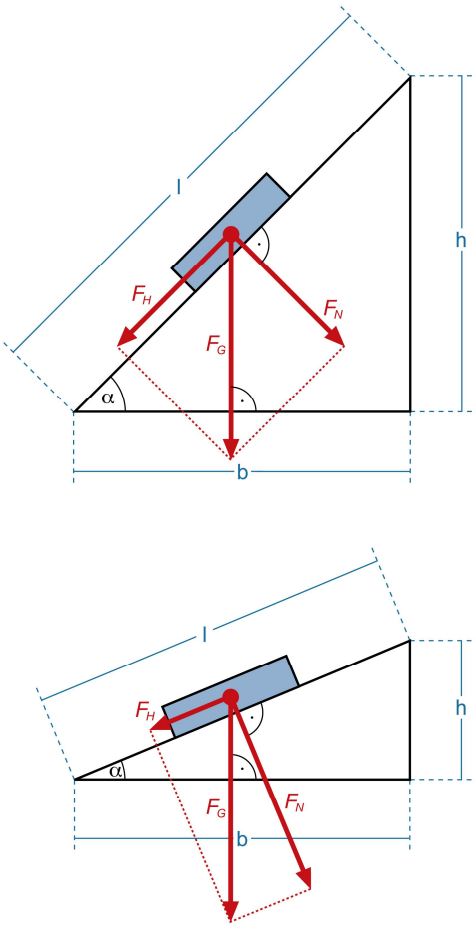


Abb. 2: Hangabtriebskraft (Quelle: Wikipedia [3])

Durch die Wahl des Neigungswinkels des Tragseils kann ein Kompromiss zwischen der auf das Zugseil wirkenden Kraft und der am Tragseil aufliegenden Kraft gefunden werden. Z. B. ist bei einem Winkel von 45° (also 100% Steigung) die senkrecht zum Tragseil wirkende Kraft genau gleich groß wie die Kraft am Zugseil (die Hangabtriebskraft), nämlich je die Gewichtskraft dividiert durch $\sqrt{2}$.

Zu Beginn hatte ich eine wesentlich größere Kabine mit der Statik-Grundplatte 90 · 180 mm gebaut, doch diese war einfach zu schwer und zudem ein Hindernis für die Einfahrt bei der Bergstation. Schließlich habe ich mich für eine ultraleichte Kabine mit 90 · 120 mm Grundfläche entschieden, welche fast nur aus Statikteilen besteht. Sie bringt ohne das Laufwerk gerade mal 190 g auf die Waage.

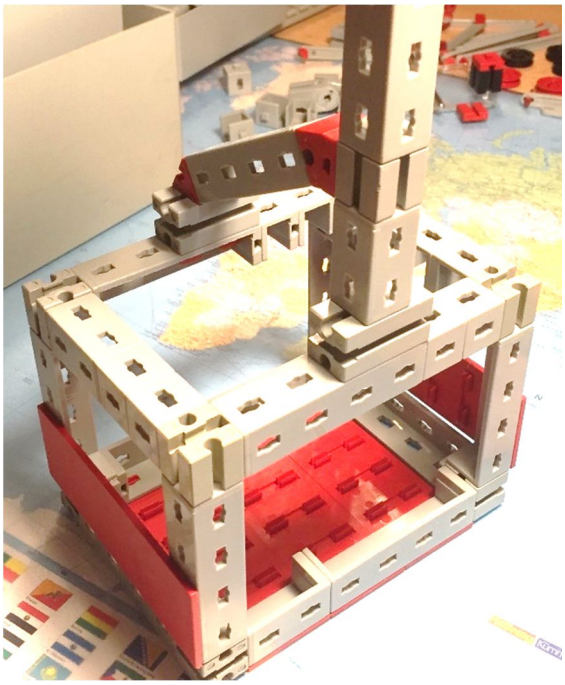


Abb. 3: Die Kabine (ohne Abdeckungen)

Die Bergstation ist mehrfach verstrebt und abgespannt. Da ich zu Beginn noch nicht wusste, welche Steigung ich wählen würde, ist sie relativ hoch gebaut (Abb. 4).

Das Tragseil ist ein Nylonfaden. Er ist in der Talstation verankert, wird hochgeführt, geht dort über eine Umlenkrolle, wieder runter,

über zwei Umlenkspulen auf die andere Seite, wieder hoch, wieder runter und schließlich auf eine Spule mit einem kleinen Motor (Abb. 5).

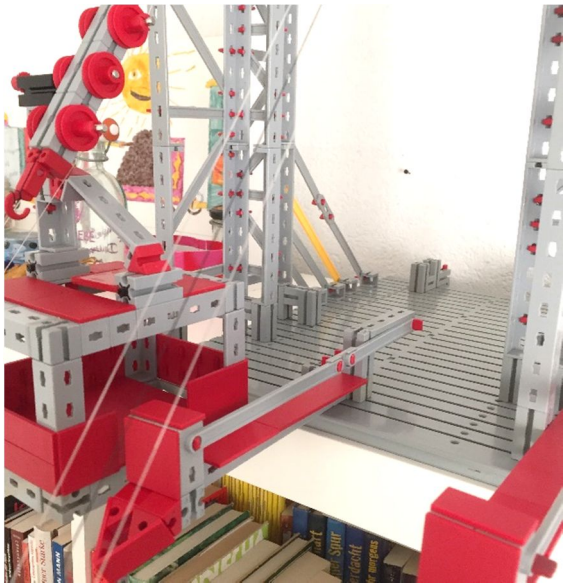


Abb. 4: Die Bergstation

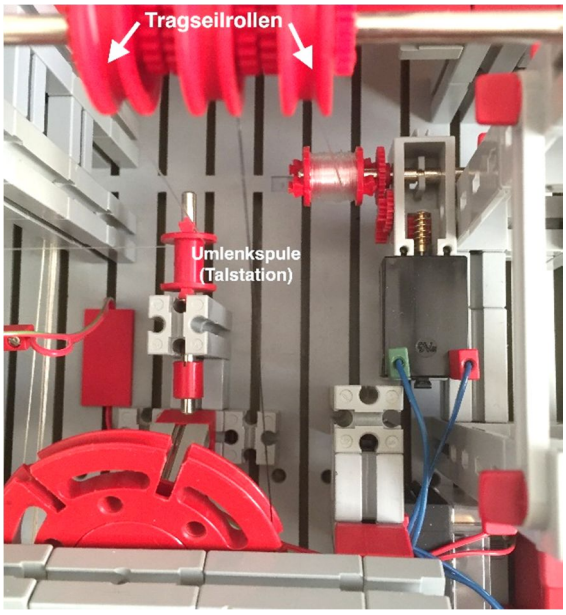


Abb. 5: Das Tragseil

Das Anziehen des Tragseils wird eher manuell durch Verschieben der Talstation bewirkt. Der Motor dient dazu, kleine Anpassungen der Spannung zu erleichtern.

In der Wirklichkeit ist das Tragseil ein Stahlseil mit mehreren Zentimetern Durchmesser (je nach Last), welches pro Kilometer einige Tonnen (!) wiegt.

Das Zugseil läuft oben und unten über große Umlenkräder, wovon in der Talstation eines angetrieben ist. Bei der linken Kabine ist es einige Male um beide Haken gewickelt, bei der rechten am oberen Haken befestigt und unten kann es mit einer Spule angezogen oder gelockert werden.

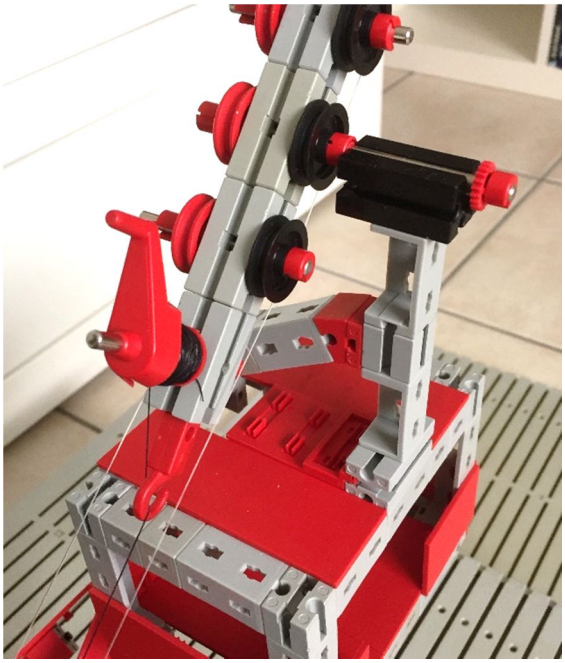


Abb. 6: Zugseil mit Spannvorrichtung

Das angetriebene Rad hat einen Gummiring, um den Faden mit der nötigen Reibung zu bewegen.

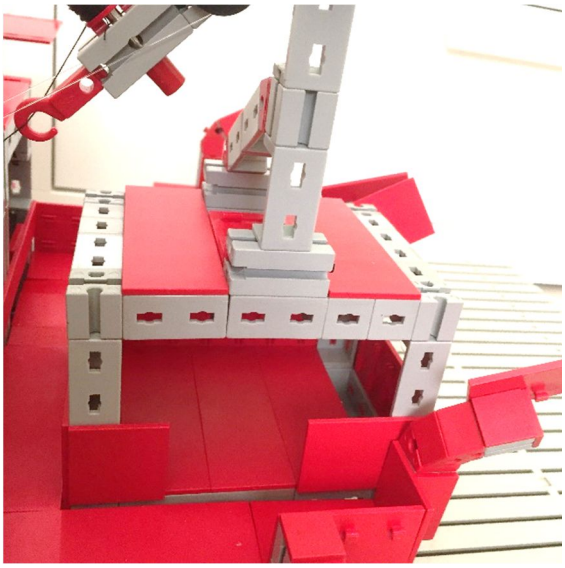


Abb. 7: Die Talstation mit stehender Kabine

Die Talstation ist ordentlich ausgebaut, die Kabinen fahren punktgenau ein, damit die Passagiere ohne Stolpern ein- und aussteigen können.

Schließlich habe ich eine Abdeckung für das Maschinenhaus der Talstation gebaut:

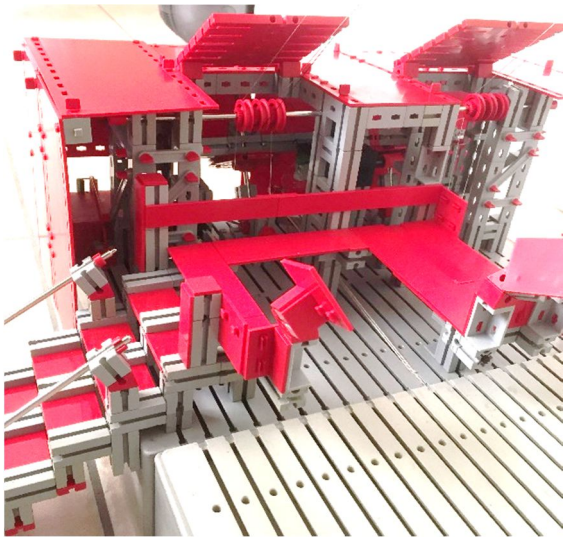


Abb. 8: Talstation mit Dach

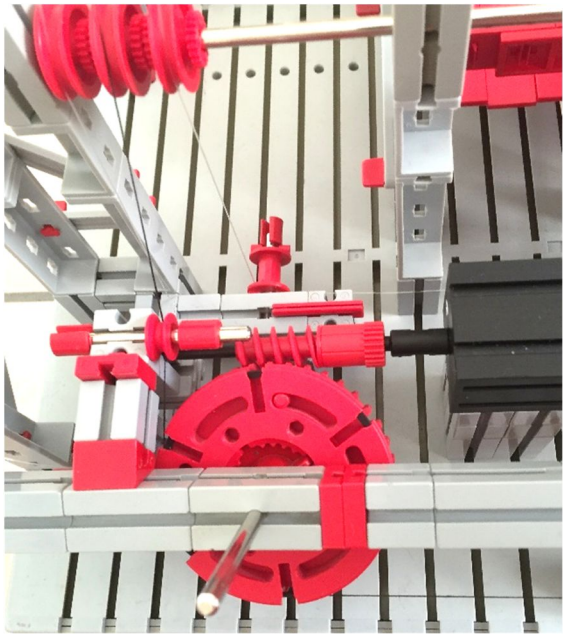


Abb. 9: Der Antrieb

Übrigens noch ein Wort zur Verankerung der Talstation: diese sollte ja nicht einfach auf dem Boden rumrutschen, sonst hängen Trag- und Zugseil bald bedrohlich durch! Ich habe mir so geholfen, dass ich die Grundplatte mit einer Schnur fest mit der gefüllten, mehrere Kilogramm schweren Sammelbox verbunden habe. Dies genügt in der Praxis, sodass sich die Talstation nicht von selber verschiebt.

Bei der Antriebseinheit läuft das Zugseil über eine zusätzliche Rolle, damit der Winkel stimmt (dass ich das Antriebsrad auch waagrecht stellen könnte, ist mir erst später eingefallen):

Einige Impressionen vom fertigen Modell:

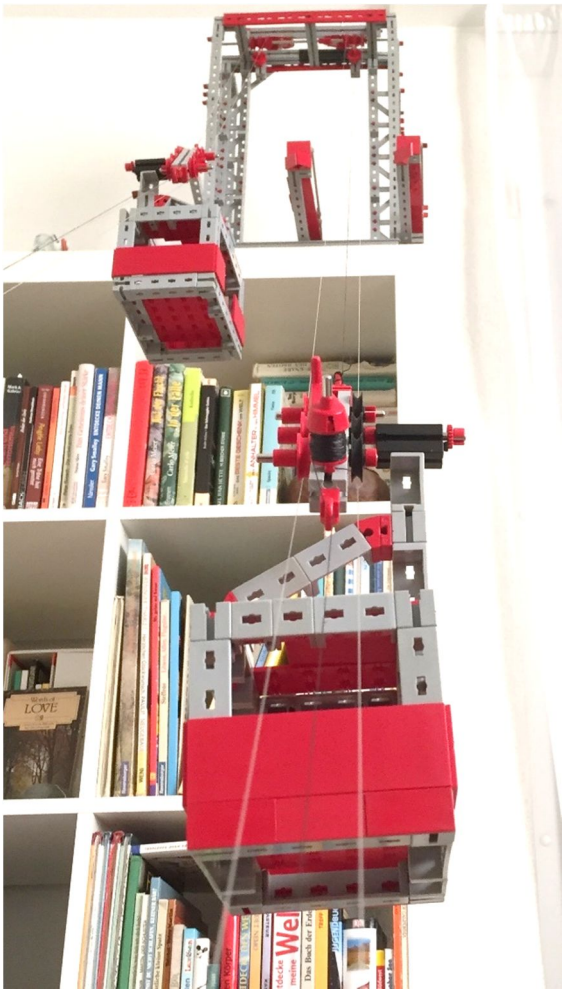


Abb. 10: Berg- und Talfahrt

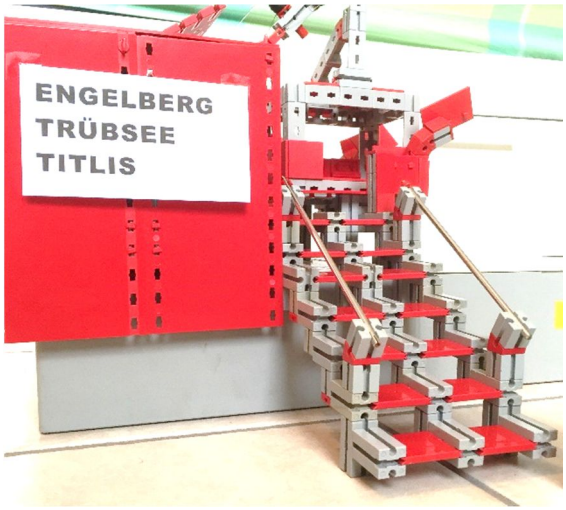


Abb. 11: Reiseziel

Und hier gibt‘s Fahrkarten:

Abb. 12: Ticketschalter

Die Führungen sorgen für eine ruhige Einfahrt auch bei böigem Wind:

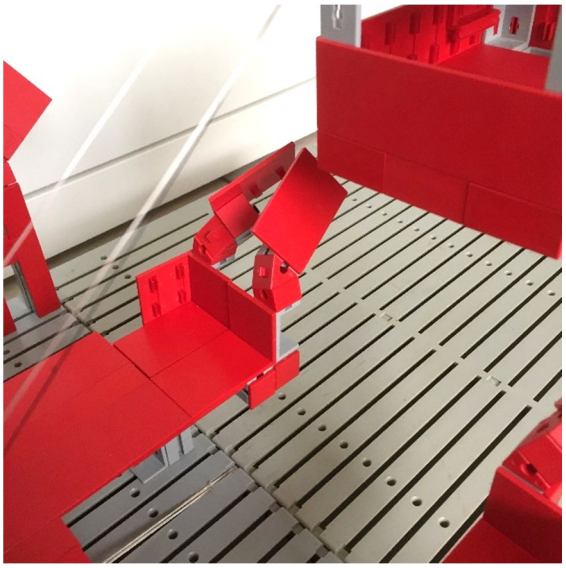


Abb. 13: Einfahrt Talstation

Quellen

[1] Wikipedia: *Luftseilbahn*.

[2] Wikipedia: *Vergleich herausragender Luftseilbahnen*.

[3] Wikipedia: *Hangabtriebskraft*.

Optik

Das Hyper-Pseudoskop

Thomas Püttmann

Dinosaurierköpfe ragen bedrohlich weit aus der Leinwand heraus, ein Blick aus dem Hochhaus nach unten löst Höhenangst aus – solche 3D-Effekte kennt man aus dem Kino. Mit nur 18 fischertechnik-Bauteilen lässt sich aber auch der Alltagswelt eine ungewohnte räumliche Tiefe verleihen oder, noch effektvoller, die räumliche Tiefenstaffelung in bestimmten Bereichen des Gesichtsfelds umkehren.

Räumliches Sehen

Das räumliche Sehen ist ein komplexer Prozess und basiert auf mehreren Effekten. Auch wenn man nur mit einem Auge schaut, erhält man schon Entfernungsinformationen. Ähnlich wie bei einer Kamera wird das betrachtete Objekt durch Veränderung der Brechkraft der Linse vom Gehirn aktiv scharf gestellt. Durch diese sogenannte *Akkomodation* kennt das Gehirn grob die Entfernung des Objekts – im Nahbereich deutlich genauer als im Fernbereich.

Weitere Informationen liefert der *Bildinhalt*. So werden die relative Größe der Objekte, Perspektivlinien, Verdeckungen, Schatten und ähnliches analysiert, um eine Tiefenstaffelung des Bildinhaltes zu erhalten.

Bewegt sich der Beobachter seitlich zur Blickrichtung, so bewegen sich die Objekte im Gesichtsfeld je nach räumlicher Tiefe unterschiedlich schnell. Jeder kennt das vom Blick aus einem fahrenden Bus oder Zug. Weit entfernte Objekte verharren an ihrer Position, während nahe Objekte vergleichsweise schnell vor ihnen her durch das Gesichtsfeld ziehen. Diesen Effekt nennt man *Bewegungsparallaxe*.

Das eigentliche räumliche Sehen erfordert den Einsatz zweier Augen. Dabei werden im Wesentlichen zwei Effekte ausgewertet, die *Konvergenz* und die *Querdisparation*.

Konvergenz

Richtet man seine Aufmerksamkeit auf ein entferntes Objekt, so sind die Blickrichtungen beider Augen parallel. Bei Entfernungen unterhalb von ca. 3 m werden die Augen nach innen gedreht, und das Gehirn kann aus der Größe dieser Auslenkung grob auf den Abstand des Objekts zurückschließen. Die Bilder auf den beiden Netzhäuten müssen vom Gehirn miteinander verschmolzen werden. Diesen Vorgang nennt man *Fusion*.

Querdisparation

In unserem Gesichtsfeld befinden sich in der Regel mehrere Objekte in unterschiedlichen Abständen. Unser Gehirn kann nicht bei all diesen Objekten gleichzeitig die von den beiden Augen gelieferten Bilder verschmelzen. Dazu ein einfacher Versuch:

Betrachte einen Gegenstand *G*. Bringe nun einen deiner Zeigefinger *Z* aufrecht zwischen dein linkes Auge und den Gegenstand. Die Situation ist in Abbildung 1 dargestellt. Der Gegenstand *G* und der Zeigefinger *Z* werden im linken Auge auf dieselbe Stelle der Netzhaut projiziert. Im rechten Auge dagegen liegt der Bildpunkt des Zeigefingers weiter rechts auf Netzhaut.

Fokussierst du nun den Gegenstand *G*, so werden die Punkte G_L und G_R vom Gehirn miteinander verschmolzen. Den Zeigefinger *Z* dagegen siehst du doppelt. Beide Kopien sind transparent, eine erscheint direkt vorm Gegenstand, die andere links davon, da der Bildpunkt Z_R auf der Netzhaut weiter rechts liegt als der Bildpunkt G_R. Beachte, dass durch die Projektion im Auge links und rechts auf der Netzhaut ebenso vertauscht werden wie oben und unten. Man nennt das Doppeltsehen des Zeigefingers *gekreuzte Querdisparation.*

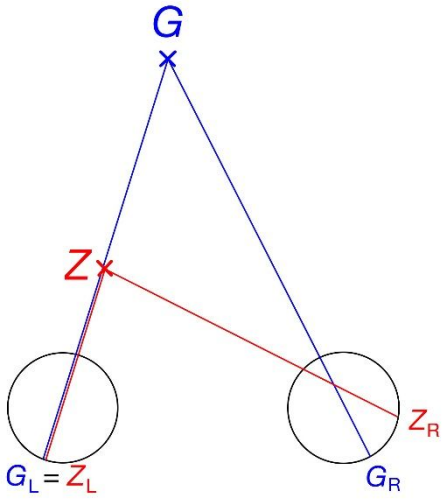


Abb. 1: Betrachtung eines Gegenstands G und eines Zeigefingers Z mit beiden Augen.

Fokussierst du nun den Zeigefinger *Z*, so werden vom Gehirn die Punkte Z_L und Z_R miteinander verschmolzen und der Gegenstand *G* erscheint doppelt. Eine Kopie liegt direkt hinter dem Zeigefinger, die andere weiter rechts. Dieses Doppeltsehen des Gegenstands nennt man *ungekreuzte Querdisparation.* Durch die gekreuzte oder ungekreuzte Querdisparation erhält das Gehirn eine sehr gewichtige Information darüber, welches Objekt vor welchem liegt und wie weit.

Das Hyperskop

Menschen mit einem größeren Augenabstand haben tendenziell eine stärkere Raumwahrnehmung, da in gleichen Situationen Konvergenz und Querdisparation größer sind als bei Menschen mit kleinerem Augenabstand. Schon *Charles Wheatstone* (1802-1875) und *David Brewster* (1781-1868) erfanden Apparate, mit denen man ausprobieren kann, wie die Welt aussieht, wenn man einen mehr als doppelt so großen Augenabstand hat. Unter dem Namen *Hyperskop* wurde ein solcher von *Terry Pope* konstruierter Apparat 1986 in der Kolumne *The Amateur Scientist* in der Zeitschrift *Scientific American* beschrieben [1].

Die Grundidee ist selten einfach: Mittels eines parallelen Spiegelpaars wird ein Auge nach außen verlagert, siehe Abbildung 2. Bei geeigneten Szenerien ist der Effekt ungemein beeindruckend – vor allem, wenn der Bildinhalt dem Gehirn keinen starken Aufschluss über die räumliche Tiefenstaffelung liefert, z. B. bei Bäumen oder Büschen mit Ästen, die auf einen zu ragen. Beim Ausprobieren kann man sehr viel darüber lernen, wann welche Tiefenwahrnehmungseffekte dominant sind.

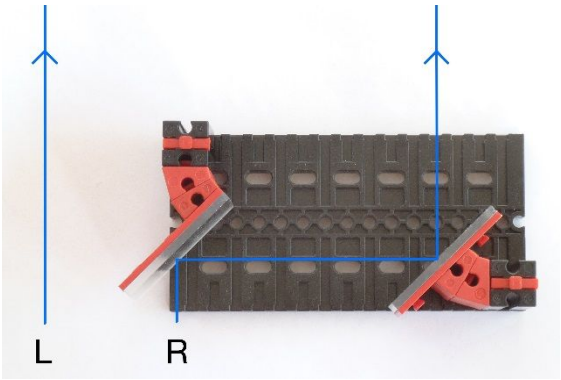


Abb. 2: Mit dem Hyperskop wird der Abstand zwischen dem linken Auge L *und dem rechten Auge* R *künstlich vergrößert.*

Das Pseudoskop

Wenn man den gleichen Apparat anders nutzt, kann man die Sehstrahlen beider Augen vertauschen. Dabei wird aus gekreuzter Querdisparation ungekreuzte und umgekehrt. Auch die Konvergenz ist entgegengesetzt der des normalen Blicks. Wenn Konvergenz und Querdisparation die dominanten Effekte sind, kehrt sich damit die Tiefenwahrnehmung um. Besonders

geeignet sind dazu Ansichten ohne viele Fluchtpunktlinien. Wenn man einzelne Büsche oder Bäume vor einem Wald betrachtet, scheint der Wald wie ein schleieriger Vorhang vor den Büschen oder Bäumen zu schweben. Eine runde Küchenuhr an der Wand scheint bei gerader Draufsicht in die Wand eingelassen zu sein.

Beeindruckend sind auch Objekte wie die Armillarsphäre aus [2], bei denen sich die wenigen Hinterschneidungen komplett ins Gegenteil verkehren. Es dauert eine gewisse Zeit, bis diese Effekte eintreten. Hilfreich ist dabei, wenn man sich nicht zu konzentrieren versucht, sondern möglichst weit abschaltet.

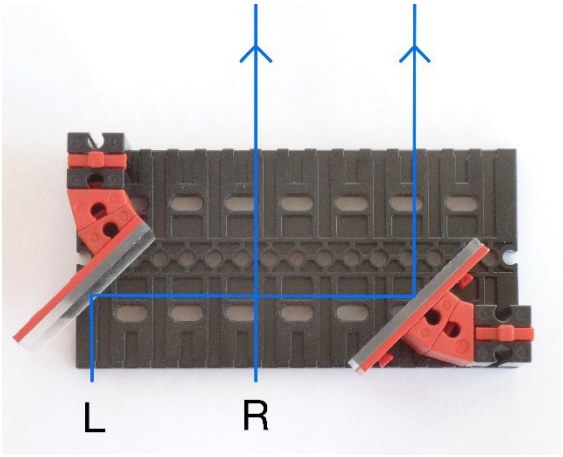


Abb. 3: Bei der Verwendung als Pseudoskop werden die Sehstrahlen der beiden Augen vertauscht.

Weitere interessante Experimente mit dem Pseudoskop findet man in [1] und [3]. Am meisten Spaß macht es natürlich, selbst auf Entdeckungstour in einer fremden Wahrnehmungswelt zu gehen. Aber Vorsicht: Das visuelle System hat gut zu tun, die sich zum Teil widersprechenden Effekte zu interpretieren. Durch langen Gebrauch des Hyper-Pseudoskops wird man daher ziemlich müde.

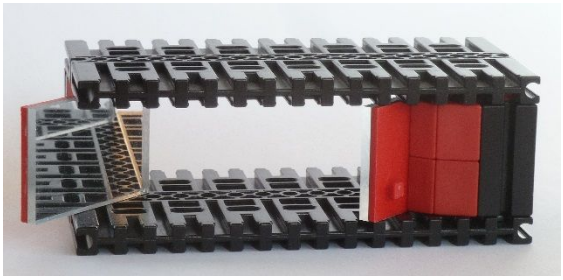


Abb. 4: Das komplette Hyper-Pseudoskop aus anderer Perspektive.

Literatur und Links

[1] Jearl Walker: *The Amateur Scientist: The hyperscope and the pseudoscope aid experiments on three-dimensional vision*, Scietific American, November 1986, online abrufbar unter [3]. Deutsche Übersetzung: Spektrum der Wissenschaft, Januar 1987.

[2] Dirk Fox, Thomas Püttmann: *Technikgeschichte mit fischertechnik*, dpunkt.verlag, Heidelberg, 2015.

[3] Terry Pope: *Phantascope*, http://www.phantascope.co.uk.

Mechanik

Verstellbare Propeller

Harald Steinhaus

Windkraftanlagen, Propellerflugzeuge und die meisten Schiffe müssen mit wechselnder Stärke der anstehenden Strömung (Luft oder Wasser) zurechtkommen und brauchen daher verstellbare Propeller, d. h. solche, bei denen man den Anstellwinkel der Blätter verändern kann. Ein Hubschrauber in der meist verwendeten Bauweise löst das gleiche Problem gleich zweimal: der Heckrotor als Ganzes und die kollektive Blattverstellung am Hauptrotor funktionieren als verstellbare Propeller. Die zyklische Blattverstellung eines Hubschraubers fügt noch eine weitere Komponente hinzu [1]. Aber selbst wenn man im Umkehrschluss annimmt, dass ein Verstellpropeller leichter zu bauen ist, steht man immer noch vor einem nicht so trivialen Problem. Ja, wenn denn fischertechnik eine Taumelscheibe als fertiges Bauteil hätte... So aber muss man auf sperrig bauende Alternativen ausweichen, die sich auch wenig für realistische Drehzahlen und die damit verbundenen Fliehkräfte eignen.

Die hier vorgestellten Propeller fallen in zwei Kategorien:

- Zum einen ist da die Bauart, die sich aus der Taumelscheibe eines Hubschraubers ableitet und nur die kollektive Blattverstellung ermöglicht, also salopp gesagt, nur die Scheibe, aber nicht das Taumeln verwendet. Das Problem hierbei ist es, ein Teil zu finden, das lose auf einer Achse dreht, an mindestens einer Seite eine umlaufende Krempe aufweist, an der man es längs verschieben kann, und das zur anderen Seite hin die Befestigung von Blattverstellhebeln ermöglicht.
- Die zweite Kategorie kommt ohne Taumelscheibe aus, indem der Anstellwinkel der Blätter durch die Drehbewegung einer zentralen Achse relativ zum Propellerkopf eingestellt wird. Anders ausgedrückt: die Blattverstellung erfolgt entweder durch Längsverschiebung oder durch Verdrehen des Propellerkopfs relativ zur antreibenden Achse.

Eine dritte, und zwar die in der Technik vorherrschende Kategorie muss leider unbestückt bleiben: die hydraulische Blattverstellung benötigt hohlgebohrte Wellen und Drücke, die mit fischertechnik-Teilen nicht zu bewältigen sind.

Bevor es richtig losgeht, werden zunächst die „Kaulquappen“ vorgestellt, auf die weiter unten zurückgegriffen wird:

Der fischertechnik-Spurkranz (36331) kann an seinem schmalen Rand mittels dreier Verbindungsstopfen (32316) an der Drehscheibe befestigt werden.

Bild 1: Spurkranz und Drehscheibe

Die langen S-Riegel 8 ermöglichen es, auf der ft-Drehscheibe vier Teile in 90° Abstand anzuordnen.

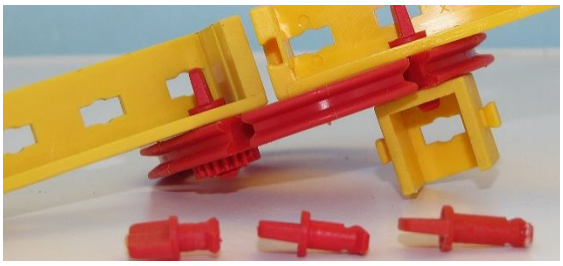


Bild 2: S-Riegel 8 und Drehscheibe

Lässt man von einer Alu-Schiene die Zapfen weg, hat man ein fischertechnik-Teil mit vier Nuten und durchgehender Bohrung.

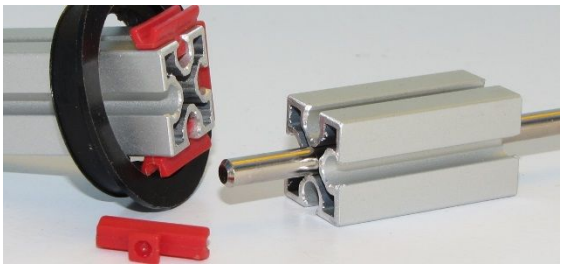


Bild 3: Alu-Schiene einmal anders

Blattverstellung mittels nicht taumelnder Taumelscheibe

Modell A

Das Modell in Abb. 4 hat vier Blätter, die mittels Rastachse mit Platte (130593), einem BS15 und der Achsverschraubung (38843) auf der Achse befestigt sind. Die Taumelscheibe wird mit vier Führungsplatten (32455) gehalten. Die Blattverstellung erfolgt durch zwei mal zwei Schneckengetriebe. Die roten Schnecken klemmen auf der Achse, weil da jeweils noch ein Stück Angelschnur (fischertechnik-Original, aus der Weihnachtsdeko (48667)) hindurch gefädelt ist.

Bild 4: Modell A mit Drehscheiben

Modell B

… verwendet das Speichenrad als Taumelscheibe. Die Verstellhebel sind durch die gekrümmten Öffnungen der Drehscheibe hindurchgeführt. Die Verstellung erfolgt durch ein Hubgetriebe, mit dem das Speichenrad (auf Freilaufnabe) auf der Achse verschoben wird.

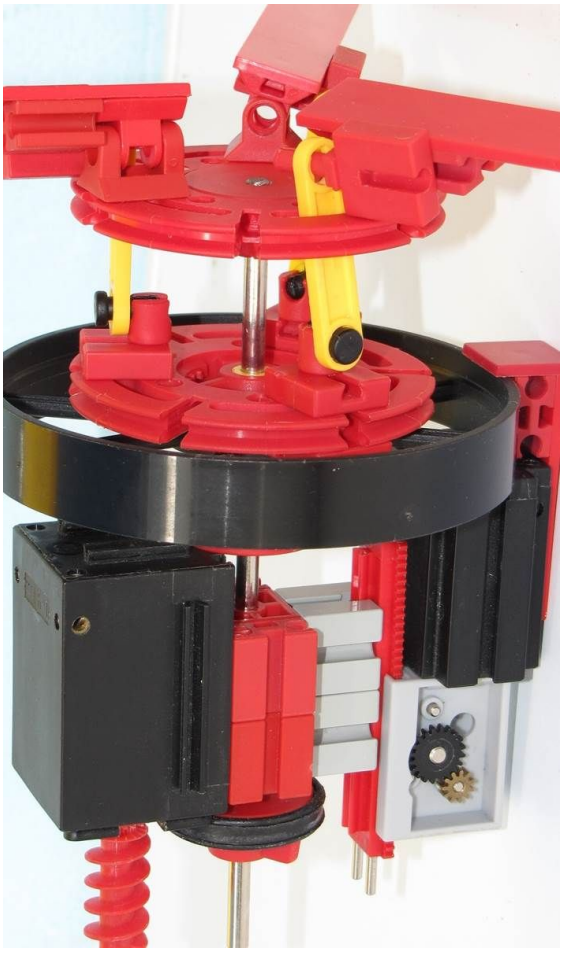


Bild 5: Modell B mit Speichenrad

Modell C

Der fischertechnik-Spurkranz (36331) ist mittels dreier Verbindungsstopfen (32316) an der Drehscheibe befestigt. Die Verstellung erfolgt mittels zweistufigem Schneckengetriebe und der Führungsplatte (32455).

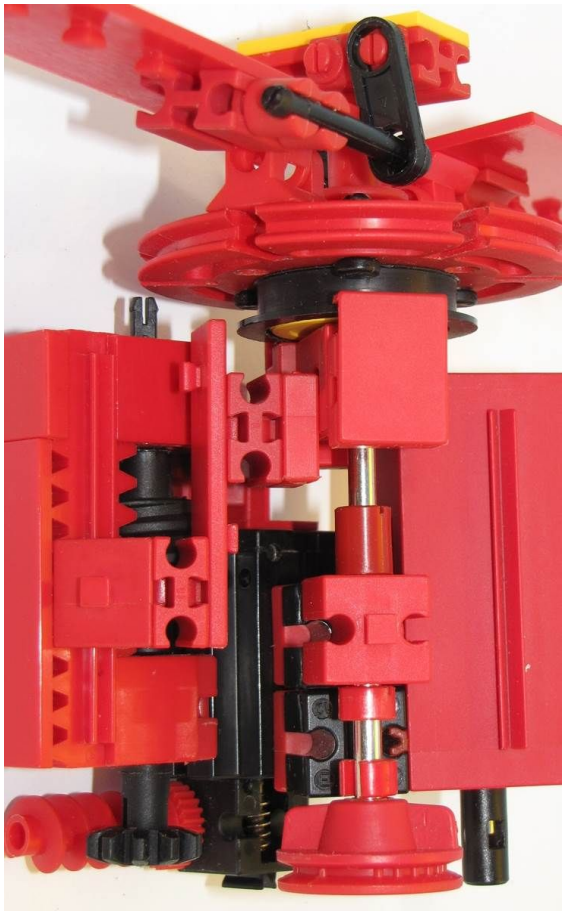


Bild 6: Modell C mit Spurkranz

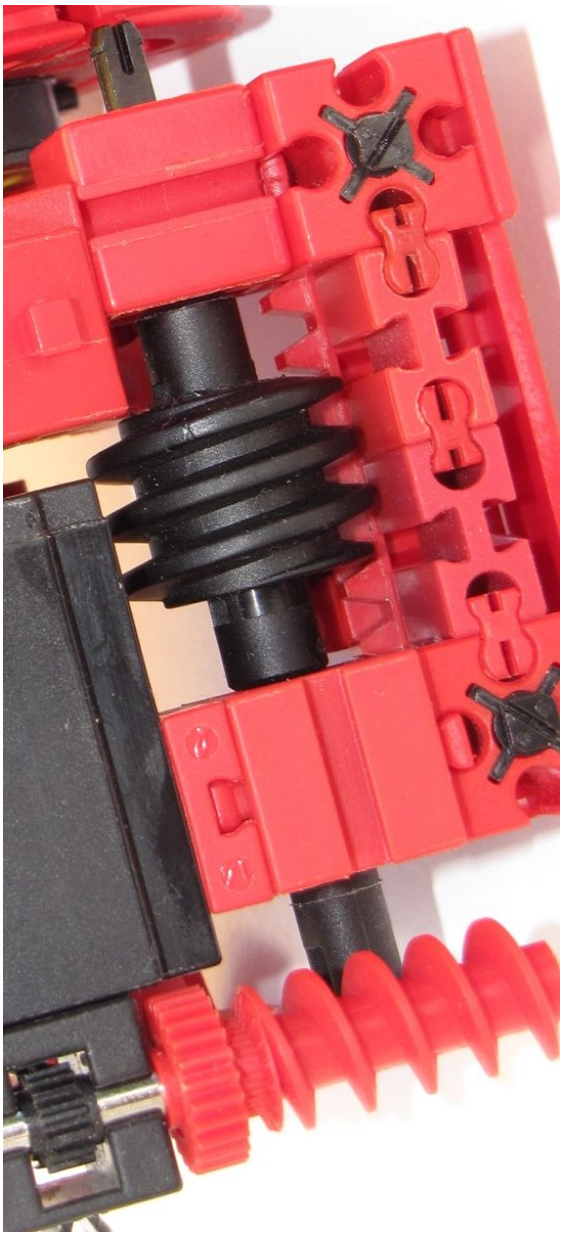


Bild 7: Modell C, Rückseite

Modell D

Die Felge 30 hat gleich zwei umlaufende Ränder, in die man Gleitführungen einhängen kann. Der Antrieb wird durch den Gummiring mehr angedeutet als tatsächlich bewirkt.

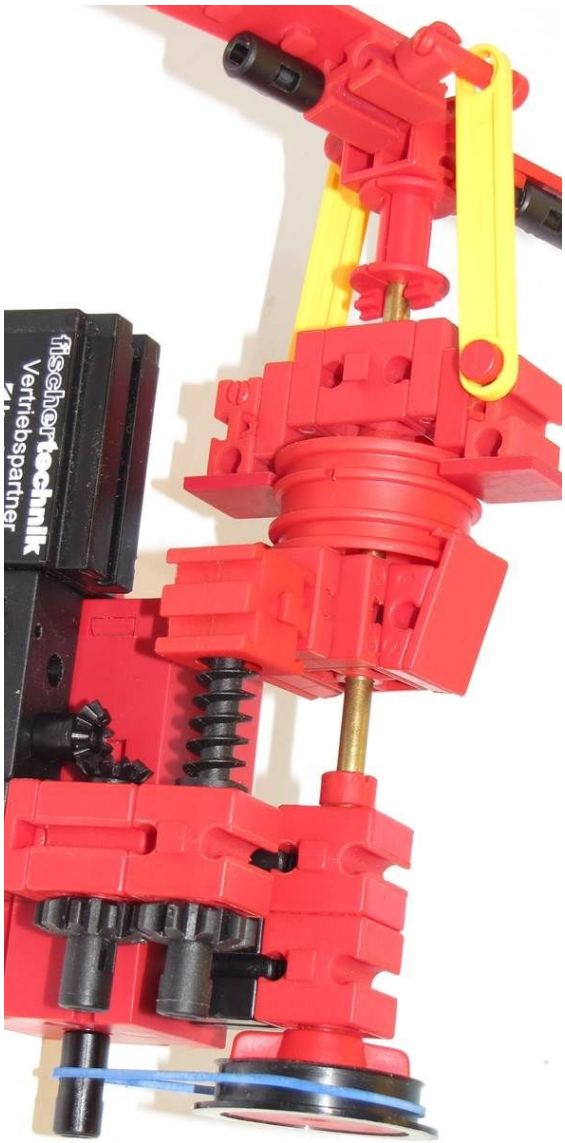


Bild 8: Modell D mit Felge 30

Modell E

Modell E verwendet ebenfalls den Spurkranz als Taumelscheibe. Als zentrales Teil im Propellerkopf dient hier ein Stück fischertechnik-Alu-Schiene ohne Zapfen, auf dem der Spurkranz mit vier Federnocken fixiert ist. Die Drehscheibe dient nur als Anlagefläche für den Spurkranz und kann noch durch etwas Kompakteres ersetzt werden.

Bild 9: Modell E mit Spurkranz und Alu

Blattverstellung mittels Verdrehen der Propellerachse

Modell F

Modell F hat drei Blätter auf einem Propellerkopf aus Drehscheibe und angekoppeltem Z30. Die Verstellung geschieht durch Verdrehen der Achse mit dem fischertechnik-Kegelzahnrad gegenüber dem Propellerkopf, auf dem die Blätter mit Differenzial-Antriebsrad (31414) montiert sind. Im Betrieb rotieren beide Elemente. Die Blattstellung bleibt genau dann unverändert, wenn sich die Elemente gleich schnell drehen. Abweichungen der Drehzahlen führen zu anderen Blattstellungen. Dieser Propeller benötigt also ein Getriebe, das für gleiche Drehzahlen sorgt, aber auch gezielt Unterschiede erzeugen kann. Dieses Getriebe kennen wir als „Gleichlaufgetriebe“, und davon gibt es im Bilderpool eine wohlbestückte Kategorie [2].

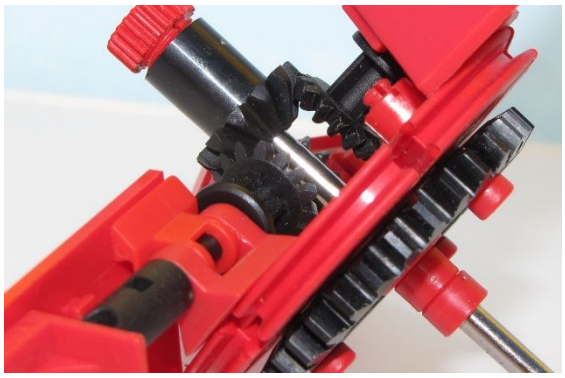


Bild 10: Modell F mit Kegelrädern

Modell G

Bild 11: Modell G mit Differenzial und Gleichlaufgetriebe

Modell G zeigt das Gleichlaufgetriebe mit einem weiteren Propellerkopf, der etwas komplexer aufgebaut ist: die Blattverstellung geschieht durch Differenzial-Antriebsräder (31414) und den Kranz des Differenzialgehäuses. Das Differenzial selbst ist am vorderen Ende blockiert, steckt mit seinem Gehäuse „einfach so“ zwischen den beiden Drehscheiben und sitzt mit dem anderen Ende auf der Antriebsachse. Dadurch ergibt sich eine Untersetzung von 1:2, wenn man die Antriebsachse gegenüber dem Propellerkopf (mit den beiden Drehscheiben) verdreht.

Beim Gleichlaufgetriebe wird der Eingang für gleichsinnigen Antrieb mit einem Power-Motor bestückt; der gegensinnige Antrieb geschieht über ein Schneckengetriebe.

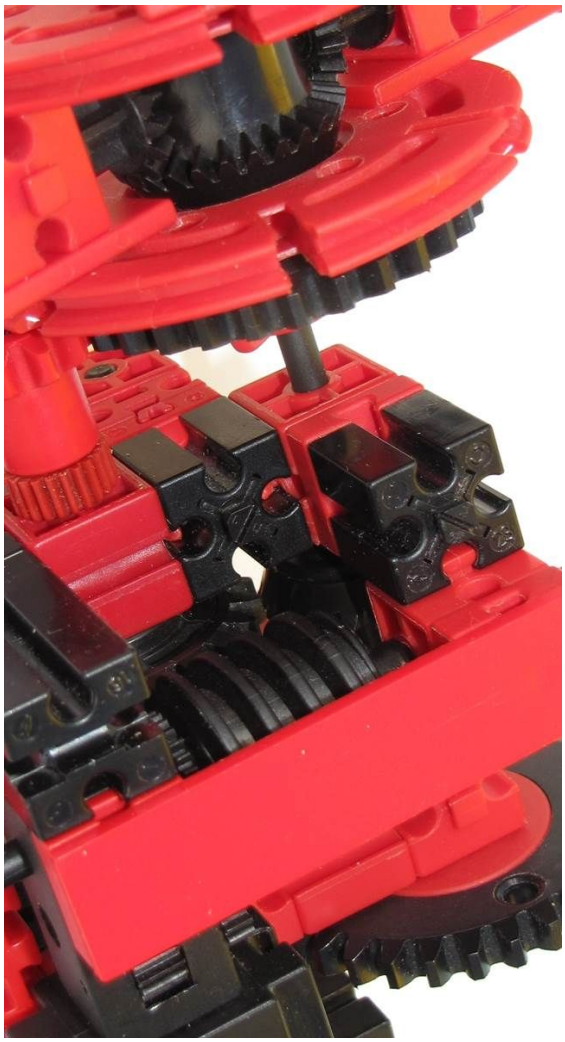


Bild 12: Modell G, Rückseite mit gegensinnigem Antrieb

Quellen

[1] Dirk Fox, Johann Fox: *Hubschrauberrotoren*. ft:pedia 3/2011, S. 29-35.

[2] ft-Community-Bilderpool: *Gleichlaufgetriebe*.

Elektromechanik

Der etwas andere Motor

Rüdiger Riedel

Die Konstruktion von Elektromotoren mit fischertechnik hat schon immer fasziniert. Seit der Verfügbarkeit starker Neodym-Magneten eröffnen sich dafür ganz neue Möglichkeiten [1, 2, 4]. Die Funktionsweise des in diesem Beitrag vorgestellten Motortyps ist besonders überraschend – auch wenn der Motor selbst fast kein Drehmoment erzeugt…

Hintergrund

Der im Folgenden vorgestellte Motor ist ein Motortyp ohne jede praktische Bedeutung, für einen Hobby-Techniker jedoch umso interessanter.

Als Kind habe ich Ende der fünfziger, Anfang der sechziger Jahre alles Mögliche gebastelt und ausprobiert. Die Modellflugzeuge von Faller im Maßstab 1:100 waren preislich in meiner Taschengeld-„Liga“ und nahmen auch wenig Platz ein in meinem Kinderzimmer. Da ich mich immer für alles Elektrische begeisterte, erstand ich auch den „Faller Elektro-Einbaumotor FM 1001“. An Wechselstrom angeschlossen machte dieser Geräusche wie ein winziger Kolbenmotor; ein aufgesteckter Propeller drehte sich sehr schön, entwickelte aber kein nennenswertes Drehmoment.

Meiner Erinnerung nach lag die Drehzahl des Propellers bei 5-10 Umdrehungen pro Sekunde oder 300-600 Umdrehungen pro Minute. Den Propeller konnte ich festhalten ohne eine besondere Kraftwirkung zu spüren; nach der Beschreibung war das für den Motor auch unschädlich. Die Dimensionen kenne ich nicht mehr, schätzungsweise lagen sie bei 2,5 bis 3 cm Länge und 0,6 cm Durchmesser.

Als „kleiner Forscher“ musste ich diesen Motor natürlich eines Tages auseinander nehmen, um seiner Funktionsweise auf die Spur zu kommen. Wie ein herkömmlicher Gleichstrommotor funktionierte wusste ich bereits, was aber hier zum Vorschein kam blieb mir damals ein Rätsel:

- Der „Rotor“, auf den der Propeller gesteckt wurde, bestand lediglich aus einer Stahlwelle.
- Dieser Rotor bewegte sich in einem Kunststoffröhrchen, das von einer langen Spule umgeben war.
- Am anderen Ende befand sich ein Ferritmagnet mit zwei „Polschuhen“, nämlich zwei Blechen, die über das Wellenende reichten.

Das war alles.

Nach meinen damaligen Kenntnissen konnte die durch den Dauermagneten magnetisierte Welle innerhalb der Spule nur hin und her vibrieren, sich aber nicht drehen. Was sie aber sehr wohl tat.

Erklärung

Wenn die Welle am Boden des Röhrchens aufliegt und seitlich schwingt, erhält sie eine hin und her gehende Drehbewegung. Ist die Reibung im Röhrchen an einer Stelle etwas erhöht, kann die Welle dort aufsteigen und bei der Schwingung in die andere Richtung wieder abwärts wandern. Damit ist eine volle Drehung ausgeführt. Durch

die weiteren Schwingungen und die Reibung kann die Drehung dauerhaft fortgeführt werden.

Die Drehzahl hängt dann ab vom Wellendurchmesser, dem Durchmesser des Röhrchens und der „Wackelfrequenz“.

Mit einem kleinen Modell kann man sich die Funktionsweise klar machen. Das von der Spule umgebene Röhrchen sei ein Innenzahnrad 30, die Welle ein Z20 (Abb. 1, 2).

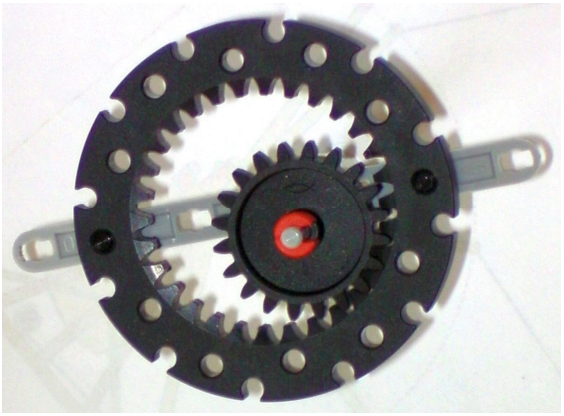


Abb. 1: Funktionsmodell, Vorderseite

Abb. 2: Funktionsmodell, Rück- bzw. Antriebsseite

Die Vibration der Welle (Wackelbewegung) ist in diesem Modell durch den Kontakt der Welle mit dem Rohr schon übergegangen in eine kreisförmige Bewegung. Die Drehung der Kunststoffkurbel 40 (38445) simuliert die 50 Hz Netzfrequenz. Das Innenzahnrad 30 wird festgehalten, dann dreht sich das Z20 (die Welle) mit halber Kurbeldrehzahl (Netzfrequenz) in entgegengesetzter Richtung.

Der im Modell feste Zusammenhang der Drehrichtungen Kunststoffkurbel und Z20 ist bei dem Motor nicht gegeben. Vielmehr kann er sowohl rechts- als auch linksherum drehen, abhängig von kleinen Unsymmetrien im Aufbau oder Zufälligkeiten beim Einschalten.

In unserem Motor liegt der „Rohrdurchmesser“ bei 6 mm, der Wellendurchmesser bei 4 mm. Somit ist eine Drehzahl von

$$3000\,\frac{U}{min}\cdot\frac{6-4}{4}=1500\,\frac{U}{min}$$

zu erwarten. Nach Augenschein dreht der Motor langsamer; gemessen habe ich es noch nicht.

Mit einer Spule aus einem Tür-Gong habe ich einen weiteren Versuch unternommen. Der Rohrdurchmesser ist ca. 5 mm, die Welle wieder 4 mm, somit komme ich auf eine Drehzahl von

$$3000\,\frac{U}{min}\cdot\frac{5-4}{4}=750\,\frac{U}{min}$$

Diese errechneten Drehzahlen nennt man auch *Synchrondrehzahlen*. Die tatsächliche Drehzahl ist kleiner oder höchsten gleich diesem Wert. Den Wert

$$\frac{Synchrondrehzahl-tats.Drehzahl}{Synchrondrehzahl}$$

nennt man *Schlupf*.

In der Praxis sind weder Röhrchen noch Welle mit Zähnen besetzt, sondern die Kraftübertragung erfolgt durch Reibung. Dadurch ist die tatsächliche Drehzahl zum Teil deutlich geringer als die Synchrondrehzahl und kann sich im Betrieb auch ändern.

Der Bewegungsablauf erinnert an die Hui-Maschine. Bei dieser wird mit einem Stöckchen über einen geriffelten Stab gestrichen, an dessen Ende ein Propeller lose mit einem Nagel befestigt ist. Die Schwingungen im Stab bringen den Propeller deshalb zum

Drehen, weil durch seitliche Krafteinwirkung (Finger) aus den ebenen Schwingungen kreis- bzw. ellipsenförmige geworden sind. Eine ausführliche Beschreibung der Funktion der Hui-Maschine findet man im Internet [2].

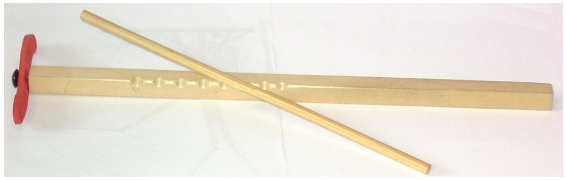


Abb. 3: Eine Hui-Maschine

Motoraufbau

Ich benutze einen Neodym-Magneten von 10 mm Länge und 4 mm Durchmesser, so einer wie in Kapitel 10 („Der Elektromotor“) von [1] beschrieben. Der Magnet haftet an einer Achse 30 als Verlängerung ohne weitere Maßnahmen.

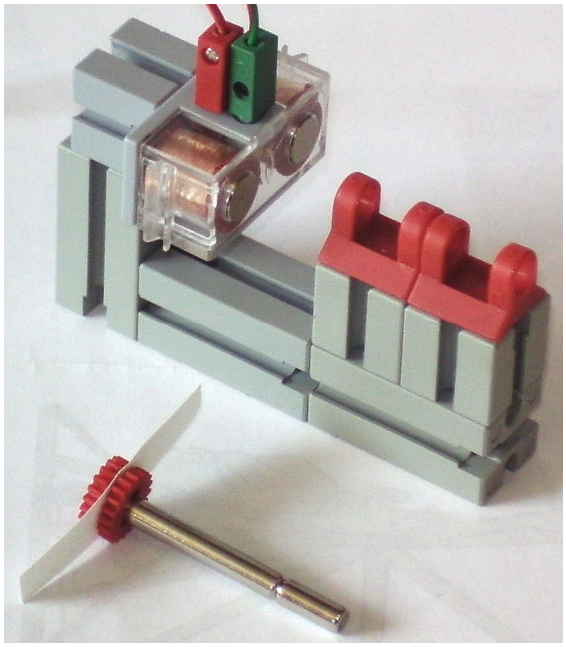


Abb. 4: Welle mit kleinem Magneten

Als Röhrchen dienen zwei Gelenkwürfel-Zungen 7,5 (31426). Der Elektromagnet wird an 6 V Wechselspannung angeschlossen. Den besten Abstand zum Neodym-Magneten probiert man aus.

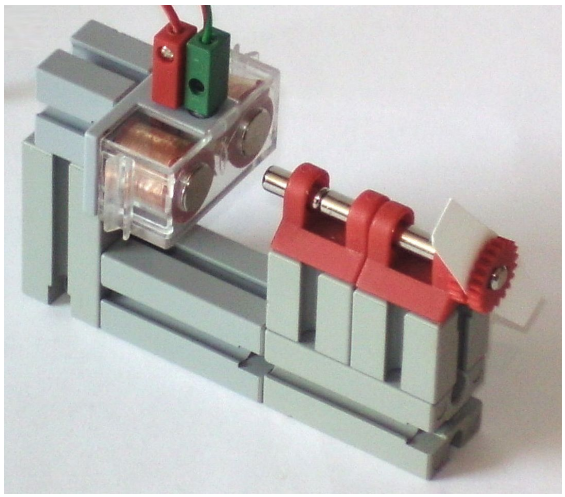


Abb. 5: Ansicht des betriebsbereiten Motors

Synchronmotor

In der ft:pedia 2/2016 schrieb Matthias Dettmer, dass Synchronmotoren mit voller Drehzahl (3000 U/min, Polpaarzahl zwei) modellmäßig mit einem Baukasten kaum zu realisieren sind. Das ist richtig. Eine Ausnahme habe ich aber gefunden: Durch die Verwendung eines „diametral“ magnetisierten Stabmagneten mit 4 mm Durchmesser und 10 mm Länge (Abb. 6).

Abb. 6: Diametral magnetisierter Stabmagnet

Setzt man diesen Magneten anstatt des bisherigen ein, dann wird die Welle durch die Wackelbewegungen in die Synchrondrehzahl von 3000 U/min geschleudert.

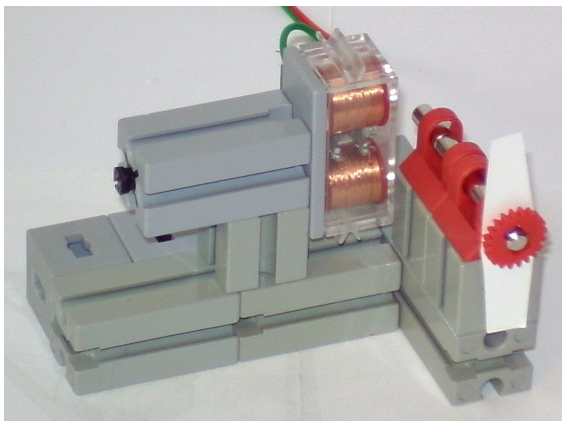


Abb. 7: Synchronmotor mit 3000 U/min

Das geringe Trägheitsmoment ist dabei entscheidend. Die Funktion wird zuverlässiger, wenn der Elektromagnet seitlich angebracht ist wie auf Abb. 7 und 8 zu sehen.

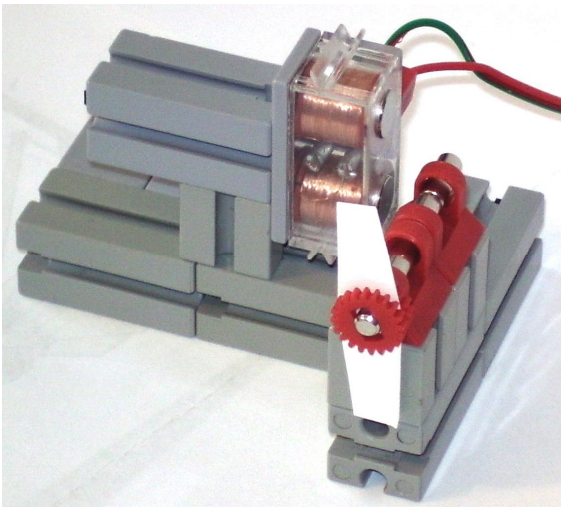


Abb. 8: Synchronmotor mit 3000 U/min

Referenzen

[1] Dirk Fox, Thomas Püttmann: *Technikgeschichte mit fischertechnik*. dpunkt.verlag, Heidelberg, 2015.

[2] H. Joachim Schlichting, Udo Backhaus: *Zur Physik der Hui-Maschine*. Physik und Didaktik 16/3, 238 (1988).

[3] Matthias Dettmer: *Synchronmotoren*. ft:pedia 2/2016, S. 48-52.

[4] Dirk Fox: *Der Elektromotor*. ft:pedia 3/2013, S. 4-8.

Grundlagen

Wecke den Erfinder in Dir

Ralf Geerken

fischertechnik ist ein Baukastensystem, mit dem sich fast jede Idee eines technischen Systems prototypisch als Modell realisieren lässt. Was aber, wenn nicht die Bauteile, wohl aber die Ideen knapp sind?

Im Frühjahr dieses Jahres postete jemand in unserem Forum folgende Frage:

Hat jemand eine Idee, was man programmieren könnte? Ich hoffe, dass es ein Paar Leute gibt, die gute Ideen haben. Und auch Ideen für Anfänger und Fortgeschrittene sind herzlich willkommen.

Ich fühlte mich jetzt nicht in der Art angesprochen, dass ich ihm der große Ideengeber sein wollte, also antwortete ich ihm:

Die besten Ideen entspringen immer noch dem eigenen Kopf. Schaue Dich einfach in deinem Umfeld um, suche nach Sachen, Dingen, Umständen, Schwierigkeiten, die dich stören, Dir missfallen oder die Du verbessern bzw. verändern möchtest. Versuche dann diese Probleme zu bearbeiten. In der heutigen Zeit ist das ja oft mit Software-Entwicklung verbunden. Ich wünsche dir jedenfalls viel Erfolg beim „Ideen-Haben“!

Motivation oder Demotivation

Was bei ihm daraus geworden ist, weiß ich zwar nicht, aber mir schwirrte das immer mal wieder im Kopf herum. Wie kommt es denn nun zum Ideen-Haben, zur Kreativität bzw. zum Erfinden?

„Weiß ich auch nicht“ werden jetzt viele sagen. Ideen hat man doch einfach so! Da bin ich aber anderer Meinung. Zur richtigen Kreativität oder gar zum Erfinden gehört ein wenig mehr, als nur Ideen zu haben. Eine Idee allein führt nämlich noch nicht zu einer Erfindung. Für die Umsetzung einer Idee in die reale Welt braucht es oft handwerkliches Geschick, Materialien zur Gestaltung, öfters programmiertechnisches Know-How, eine Entwicklungsumgebung, Zeit, Freiraum und eine gehörige Portion Mut. Oftmals werden gerade in den jungen Jahren eines Erfinderdaseins die Ideen mit den Worten *„Das ist ja völlig abgedreht!“, „Das geht doch gar nicht!“, „Das ist doch brotlose Kunst!“* oder *„Was geht denn in deinem Kopf vor?“* abgetan. Wer von den älteren fischertechnikern hat nicht wenigstens einen dieser Sätze von seinen Eltern gehört?

Meine Reaktion auf solche Sätze war jedenfalls oft, dass ich meine Kiste mit fischertechnik unter dem Bett vorgezogen und losgelegt habe, meine Ideen umzusetzen. Ich erinnere mich noch genau an eine Szene, als ich meinem Vater sagte, dass ich gerne eine Murmelbahn bauen würde. Selbstverständlich aus fischertechnik. Ich war vielleicht so 10 bis 11 Jahre alt. Mein Vater schaute sich nur die Bauteile des Statikkastens an, nahm insbesondere die Statikbögen in die Hand und meinte: *„Mit denen geht es jedenfalls nicht, die haben ja immer Unterbrechungen in der Bahnführung.“* Davon habe ich mich zum Glück nicht demotivieren lassen, habe die Bögen mit 15er Bausteinen verbunden und als Außenkante die Statikstreben umgekehrt angesetzt. Mein Vater tat jedenfalls

erstaunt, dass ich das Problem doch noch gelöst hatte. Viel später erzählte er mir, dass er mich mit seiner Aussage lediglich motivieren wollte, eigene Ideen zu haben. Er ist Maschinenbau-Ingenieur und hätte mir damals die Lösung auch verraten können. Das hat er aber nicht getan und damit einen der Grundsteine gelegt, auf die ich heute noch baue – das Umdenken.

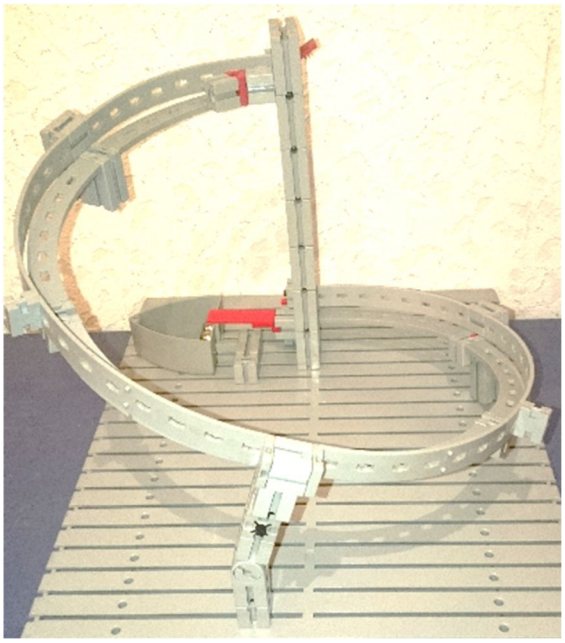


Abb. 1: So oder zumindest so ähnlich hat meine Murmelbahn von damals ausgesehen

Aus Störfaktoren werden Ideen

Grundlage für eine Idee (ob sie gut oder schlecht ist, stellt sich meist erst später heraus) ist oft eine Unzulänglichkeit in der direkten Umgebung. Irgendetwas stört oder fehlt. Manchmal muss man dann ganz genau hinschauen, was das denn ist. Da fängt das Ideen-Haben meines Erachtens nach an: Mit einer sensiblen Wahrnehmung der Umwelt. Jetzt braucht es noch einen Schuss Kreativität sowie das geeignete Material, und die Erfindung ist nicht mehr weit entfernt. Das fischertechnik-System eignet sich hervorragend für eine Umsetzung vieler Ideen. Nicht nur im Kindes-, sondern auch im Erwachsenenalter hält dieses Baukastensystem alles bereit, was das Erfinderherz begehrt. Es sind nicht zu viele verschiedene Bauteile, aber auch nicht zu wenige. Die Bauteile sind nicht zu klein, aber auch nicht zu groß. Sie sind fast „unkaputtbar“, aber auch nicht zu fest. Sie sind flexibel einsetzbar, haben aber trotzdem eine erkennbare Funktion. Das prädestiniert diese Bauteile dafür, dass selbst ein noch junger Erfindergeist schon echte Kreativität zeigen kann.

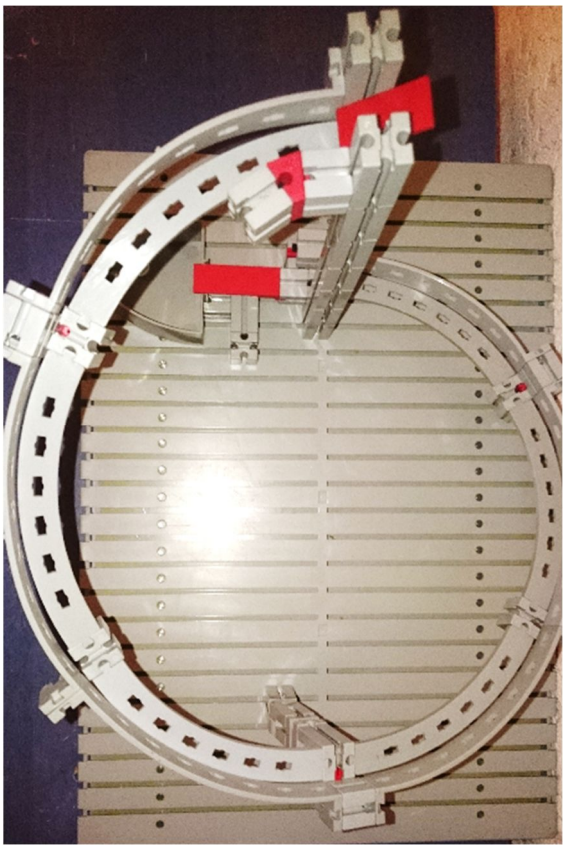


Abb. 2 Ansicht von oben

Ordnung oder Unordnung

Zuviel Ordnung verhindert so manche kreative Idee. Manchmal sind es genau die kleinen Unordnungen, die eine kreative Idee zur Folge haben. So hörte ich letztens davon, dass die grundlegende Idee zur Konstruktion eines genauen Achtecks mit fischertechnik durch drei ineinander gehakte bzw. in sich verhedderte Bausteine kam. Vielleicht berichtet René ja selbst einmal in einem eigenen Artikel davon. Auch können Unordnungen zu gewissen Unzufriedenheiten führen, ohne die ein

Fortschritt unserer Gesellschaft viel langsamer vonstattengehen würde.

Aber Vorsicht, die Kehrseite der Medaille ist genau eine Medaillendicke entfernt. Zu viel Unordnung verhindert so manche Umsetzung einer kreativen Idee. Oder habt ihr schon einmal versucht ein Modell aus einem durcheinander gewürfelten Haufen Einzelteile zu bauen? Hierbei ist es völlig egal, ob es sich um eigene Ideen oder Kreationen handelt, oder um ein Modell aus einem fertigen Kasten. In der Anfangsphase habe ich meinen Kindern beim Zusammenbauen eines vorgegebenen Modells immer die Steine von dem nächsten Schritt der Bauanleitung aus den weißen Schalen herausgesucht und bereitgelegt. Beim Zusammensuchen der Teile verlieren die Kinder nämlich oft die Geduld, weil sie manche Teile nicht so schnell finden oder zuordnen können. Sie wollen ja den Fortschritt des Modells sehen und nicht ein tolles Teilesuchen veranstalten. So hatte ich jedenfalls auch immer etwas zu tun, wenn die Kinder mal einen neuen Kasten bekamen. Zu einem guten Arbeitsplatz gehört meines Erachtens eine gute und durchdachte Ordnung, sodass die Zugriffszeit auf die benötigten Teile oder Werkzeuge sehr klein bleibt. Damit steigt die Erfolgsquote der Ideenumsetzung jedenfalls immens.

Erfolge und Misserfolge

Da man bei der Umsetzung einer Idee immer Ausprobieren muss, welcher Weg denn gangbar ist, muss man dementsprechend viele Misserfolge haben, um vielleicht mal einen Weg herauszufinden, der dann tatsächlich funktioniert. D. h. man darf sich von Misserfolgen nicht demotivieren lassen, sondern – ganz im Gegenteil – sollten sie den jungen sowie den alten Erfindergeist stärken. Misserfolge gehören eben einfach dazu. Oftmals gehört also zur Umsetzung einer Idee eine große Portion Durchhaltevermögen.

Ideen umsetzen oder Probleme lösen – oder beides?

Ist erstmal die Idee geboren und hat vielleicht auch schon die Umsetzung in irgendeiner Art begonnen, treten fast immer sofort die ersten Probleme auf. Die Umsetzung geht jetzt in eine erste entscheidende Phase – die Probleme müssen nämlich gelöst werden. Da hilft oft das Ausprobieren von verschiedenen Möglichkeiten. Manchmal erkennt man jetzt erst die Ursache des Problems und man muss vielleicht sogar zwei, drei oder noch viel öfter einen anderen Weg einschlagen – oder sogar nochmal ganz von vorne anfangen.

Das ist mir beispielsweise bei der Entwicklung der Flechtmaschine so gegangen. Beim Lösen der Probleme kommen also wieder Ideen hoch, die wieder ausprobiert werden müssen. D. h. im Klartext: Eine Idee hat viele weitere Ideen zur Folge. Ideen umzusetzen und Probleme zu lösen gehören für mich nicht nur eng zusammen, sondern das Eine geht ohne das Andere gar nicht.

Die „Definition of Done“

Ganz wichtig bei der Umsetzung einer Idee ist die „Definition of Done“, also die Definition, wann etwas fertig ist bzw. wann die Umsetzung der Idee abgeschlossen ist. Ist die „Definition of Done“ nicht gegeben, dann läuft selbst der junge Erfinder schon Gefahr, sich zu verlieren bzw. nie fertig zu werden. Das passiert vielen Erwachsenen in der Berufswelt auch immer wieder. Deswegen wurde dieser Begriff erschaffen und wird in vielen Firmen auch so gelebt. Hierfür wird meist ein ganzes Team herangezogen, welches in einer sehr frühen Phase eines Projekts schon festlegt, wann das Projekt bzw. die Umsetzung einer Idee als abgeschlossen gelten.

Erst wenn alle Probleme gelöst, umschifft oder sehr stark minimiert wurden geht deine Ideenumsetzung dem Ende zu. Du hast jedenfalls nicht aufgegeben und sie verwirklicht.

Tipps & Tricks

fischertechnik-Nutprofile selbst herstellen (2)

Andreas Tacke

In meinem Beitrag in der ft:pedia 1/2016 [1] habe ich vorgestellt, wie sich eine fischertechnik-Nut mittels einer Fräsmaschine recht einfach herstellen lässt. Dort ist die „runde" Nut beschrieben. Heute zeige ich, wie man eine abgeflachte Nut herstellen kann…

Wie wir alle wissen, gibt es bei fischertechnik zwei verschiedene Nutprofile. Die Standard-Nut ist so beschaffen, dass sich dort auch eine Achse durchführen lässt. Es gibt aber auch noch eine abgeflachte Nut, die dort zum Einsatz kommt, wo nicht genug Platz für die runde Nut vorhanden ist.

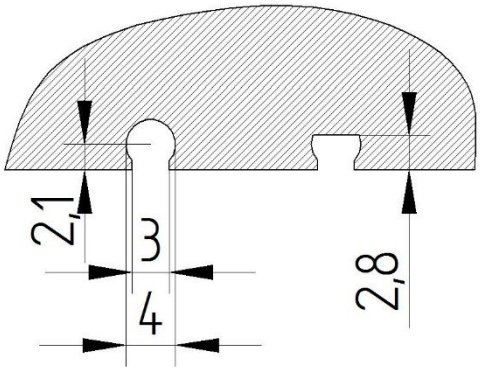


Abb. 1: Nutprofile bei fischertechnik; links die Standard-Nut, rechts abgeflacht

Nun stellt sich wieder ein Problem: Woher bekommt man den passenden Fräser? Meine Anfragen bei einigen Fräserherstellern waren nicht erfolgreich. Also galt wieder: Selbst ist der Mann…

Für die runde Nut verwende ich, wie schon in [1] beschrieben, einen Kugelfräser mit 4 mm Kopfdurchmesser. Das funktioniert mit diesem Werkzeug recht gut.

Um nun die abgeflachte Nut herzustellen habe ich so einen Kugelfräser auf einer Werkzeugschleifmaschine gekürzt. Dazu wurde der Kopf um 1,3 mm abgeschliffen. So erhielt ich einen Formfräser für die abgeflachte Nut (Abb. 2).

Abb. 2: Angepasster Kugelfräser

Zum Herstellen der Nuten werden auch hier folgende Teile benötigt:

- eine Fräsmaschine,
- ein 3 mm-Schaftfräser und
- der abgeflachte 4 mm-Kugelfräser.

Abb. 3: Fräswerkzeuge

Im ersten Schritt wird – ähnlich wie bei der Standard-Nut – eine 3 mm-Nut 2,8 mm tief gefräst (Abb. 4).

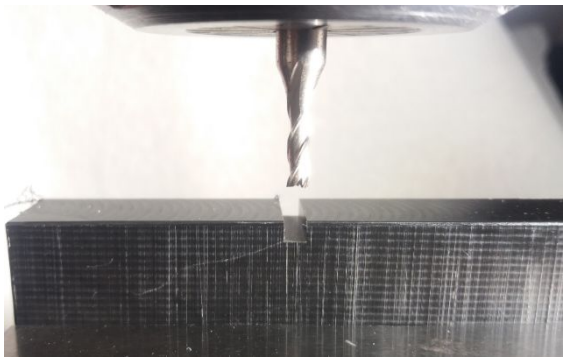


Abb. 4: Erster Schritt: Nut 3 mm, 2,8 mm tief

Im zweiten Schritt wird nun mit dem Sonderfräser das Nutprofil gefräst. Die Tiefe beträgt hierbei ebenfalls 2,8 mm.

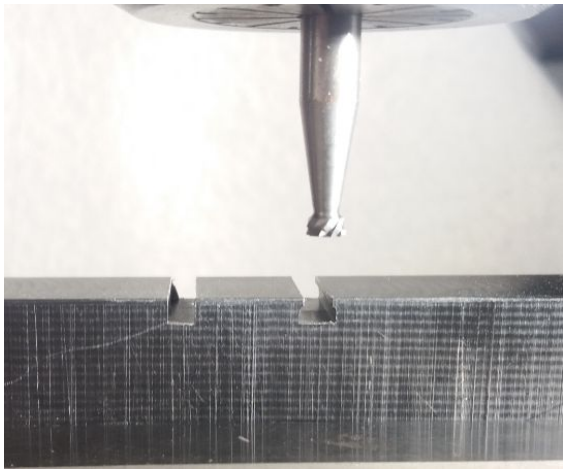


Abb. 5: Zweiter Schritt: Nutprofil fräsen, 2,8 mm tief

So entsteht das abgeflachte fischertechnik-Nutprofil, das sich mit allen fischertechnik-Teilen kombinieren lässt.

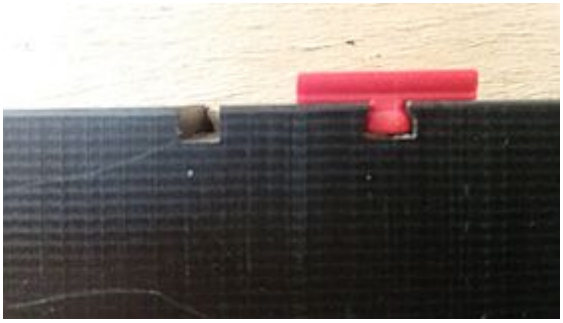


Abb. 6: Fertige Nut mit Federnocken

Für den Powermotor habe ich mir einen Adapter gebaut, bei dem ich eben diese Nut brauchte (Abb. 7).

Abb. 7: Adapterbaustein für Powermotor

Damit habe ich aufgelöst, wie ich das flache Nutprofil herstelle. Hier zeigt sich wieder, dass ein selbst gemachtes Werkzeug oft am besten funktioniert. Wer also die technischen Möglichkeiten hat, kann sich nun seine eigenen Bauteile herstellen…

Referenzen

[1] Andreas Tacke: *fischertechnik-Nutprofile selbst herstellen*. ft:pedia 1/2016, S. 8-9.

Tipps & Tricks

Tips on using the fischertechnik TXT Controller

Peter King

Did you know that you can use non-fischertechnik webcams with the TXT controller? Or that you can use the fischertechnik camera directly with Android devices? Read on to learn some details about the TXT controller that you might not have been aware of.

1. The most important requirement of the fischertechnik TXT controller (part no. 522429 or 153513) is the power supply! Always ensure that you have a charged battery or, preferably, use a mains adapter especially during important operations such as firmware upgrades. A supply voltage of 8~10 Volts is required. Important: ensure that a minimum of 8 V is maintained even when outputs are turned on!

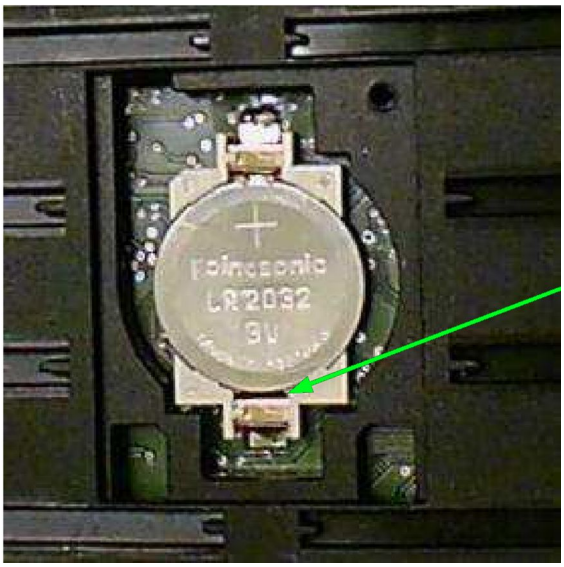
2. Should the ROBO Pro software suggest that a TXT firmware update is required, always use the USB cable connection before proceeding with the upgrade. Don't forget to switch the TXT controller off and on again after downloading the new firmware!

3. After turning the WiFi or Bluetooth on for the first time, always switch the TXT controller off and on again to ensure that any wireless operation is initialized. Also it is a good idea to key the "Pairing Code" and the "Network security key" into a TXT or DOC file on your computer to facilitate cutting and pasting when initializing the WiFi or Bluetooth connection. Please note, should your WiFi connection say "limited connectivity", simply switch the TXT controller off and on again. If the WiFi connection is working correctly, the status will show "Activity" that is both sending and receiving data!

4. When using Bluetooth on the computer, plug in your Bluetooth device or switch it on and ensure that the Bluetooth icon is visible in the notification area. Right-click on the icon and then select "Join a Personal Area Network". If it's the first time connecting the TXT to the computer, you will need to "pair" the device. Click "Add…" and select the "TXT xxxx" device, click "Next" and enter the "Pairing Code". Finally, once the device appears as an "access point" in the "Personal Area Network" window, click "connect". Please note that you should now be able to check the "Network and Sharing Center" and find that there is a "Bluetooth Network Connection" and the "Status" of this connection can be checked.

5. The backup battery compartment (pictured below) is underneath the TXT. Remove the Phillips-head screw and, using a flat screwdriver for assistance, press the latch and remove the cover. The 3 V lithium coin

battery CR2032 is used to back up the Real-Time Clock (RTC), maintaining the date and time when the TXT is disconnected from an external power source. Our tests suggest that the battery should only last 1 to 2 years. Therefore, it might be best to remove the battery if the TXT controller is not being used for a long period of time. When removing the battery, it is important not to short-circuit the contacts. Please use the procedure below and place the battery in a safe location out of reach of young children as it represents a serious health hazard if swallowed. You can measure the battery's voltage by connecting a cable to one of the I inputs (set to Analog 10V in) and touching the plugs to the battery's terminals whilst using the "Interface Test" in ROBO Pro. The battery should be replaced if the voltage is below 2.5 V (< 2500).



The TXT controller's battery compartment. Insert a strip of paper at the denoted location.

Insert a small strip of paper in the gap to act as an insulator (see picture), use a small screwdriver to lever the battery out of the battery holder. Never attempt to lever the battery out using the plastic case of the TXT controller as damage can occur!

6. To set the "Date & Time" on the TXT controller, press "Settings" and scroll down by pressing the "∇" symbol (top RH corner) and then press "Date & Time". Note the time is set as a 24-hour clock 00:00 to 23:59. The seconds are automatically reset to zero when the green tick is pressed.

7. The TXT Discovery Set (part no. 524328) includes a USB camera module which may also be purchased separately (part no. 152522). This camera with microphone may be used as a webcam on your PC or it may be plugged directly into the TXT controller's USB A socket. The big advantage of the fischertechnik camera is that it has adjustable focus (the picture above was taken with this camera). In fact, any USB camera which uses the Sonix SN9C259 image controller chip (e.g. ODROID USB-CAM 720P) can be used with the TXT controller.

The fischertechnik camera can be used with Windows, Mac OSX and Linux. The camera can also be plugged directly into Android phones and tablets using a micro USB-OTG adapter. For a suitable app, see "UsbWebCamera" [1].

8. When the camera on the TXT controller is being used (e.g. a program is running or video is being viewed) it is important that the camera should NOT be removed from the USB socket. The TXT may freeze and disconnecting the external power supply and reconnecting again is the only way to recover. Finally, do not attempt to preview the camera on the TXT controller whilst connected by Bluetooth. Bluetooth does not have the necessary data transfer rate required for camera viewing.

9. A number of TX and TXT Controller apps are available for Android devices [2].

References

[1] „Serenegiant": *UsbWeb*. Android app for USB cameras in Google Play, 2016.

[2] Search for TX fischertechnik in Google Play.

Tipps & Tricks

Große Modelle mit nur einer fischertechnik-IR-Fernsteuerung ansteuern

Dirk Wölffel

Wie steuert man komplexe mobile fischertechnik-Modelle mit nur einer fischertechnik-IR-Fernsteuerung? Spätestens bei mehr als vier Motoren oder acht Lämpchen reicht eine IR-Fernsteuerung scheinbar nicht mehr aus, um das Modell anzusteuern – aber man möchte deshalb ja nicht gleich mehrere Fernsteuerungen erwerben. Tatsächlich genügt meist eine einzige – sofern man weiß, wie …

Die IR-Fernsteuerung

Die fischertechnik-IR-Fernsteuerung ist eine 4-Kanal-Fernsteuerung. Das bedeutet, dass in der Regel vier Motoren oder acht Lämpchen angesteuert werden können. Das funktioniert mit dem neuen Robotics TXT Controller inzwischen sogar ganz ohne das IR-Empfängermodul.

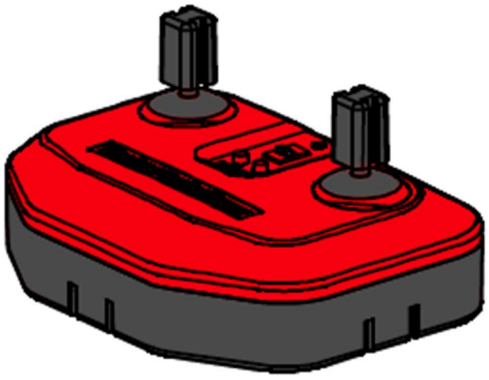


Abb. 1: IR-Sender (133053)

Mit einem kleinen Trick können wir via ROBO Pro sogar mehr als acht Ausgänge ansteuern. Dazu schauen wir uns einmal die Einbindung des IR-Senders in ROBO Pro an [1].

Abb. 2: IR-Symbol in ROBO Pro

Der IR-Eingang liefert je nach Auslenkung des Joysticks der IR-Fernsteuerung einen Wert von 0 bis 15 zurück. Durch Multiplikation mit 34 können wir so leicht die Geschwindigkeit eines Motors (mit einer Auflösung von 0 bis 510) in 16 Stufen ansteuern.

Wenn ihr mit der rechten Maustaste auf das IR-Symbol in ROBO Pro klickt (Abb. 2), dann klappt ein Menü zur Ansteuerung des IR-Senders auf (Abb. 3).

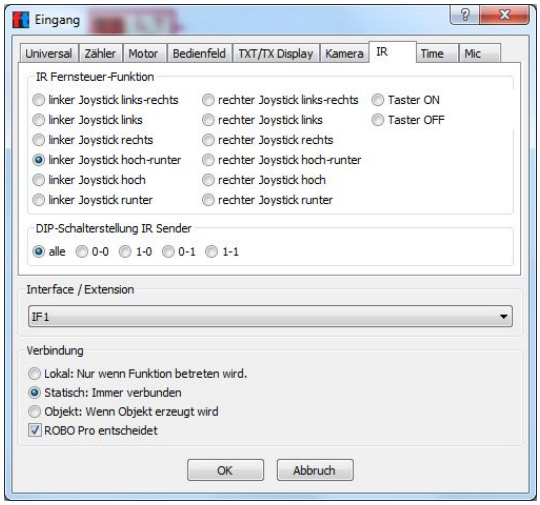


Abb. 3: Menü des IR-Senders

Hier wird die Ansteuerung für euer Modell festgelegt. Wir schauen uns nun die Auswertung der DIP-Schalterstellung des IR-

Senders in ROBO Pro etwas genauer an (Abb. 4).

Abb. 4: DIP-Schalterstellung

Wir haben jetzt vier verschiedene Möglichkeiten, den IR-Sender auszuwerten: 0-0, 1-0, 0-1 und 1-1. Für jede dieser DIP-Schalterstellungen können wir ein IR-Symbol erzeugen. Beispiel: „linker Joystick links“ (Abb. 5).

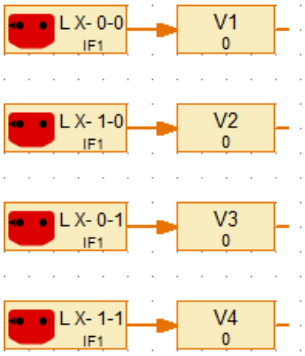


Abb. 5: „linker Joystick links“, vier Varianten

Zählt man die Taster „On“ und „Off“ hinzu, ergeben sich jetzt 40 Möglichkeiten, ein komplexes Modell anzusteuern (Abb. 6).

Dadurch haben wir die Möglichkeit, mit nur einer IR-Fernbedienung mehrere Robotics TXT Controller mit jeweils vier bzw. acht Ausgängen zu steuern – und zusätzlich das IR-Empfangsmodul. Wir müssen dazu lediglich die DIP-Schalter auf der IR-Fernbedienung umstellen.

Quellen

[1] fischertechnik: *ROBO Pro Online Hilfe, 8.7.9 IR-Eingang (TXT Controller)*.

Nr	DIP-Schalter			
	0-0	1-0	0-1	1-1
1	linker Joystick links	dito	dito	dito
2	linker Joystick rechts	dito	dito	dito
3	linker Joystick hoch	dito	dito	dito
4	linker Joystick runter	dito	dito	dito
5	rechter Joystick links	dito	dito	dito
6	rechterJoystick rechts	dito	dito	dito
7	rechter Joystick hoch	dito	dito	dito
8	rechter Joystick runter	dito	dito	dito
9	On	dito	dito	dito
10	Off	dito	dito	dito

Abb. 6: DIP-Schalterstellungen und Ansteuerungsmöglichkeiten

Computing

Codes der fischertechnik-Infrarot-Fernsteuerungen

Dirk Uffmann

Wer den Sender oder den Empfänger einer der fischertechnik-IR-Fernsteuerungen ersetzen möchte (und nicht über einen TXT verfügt), muss den von der Fernsteuerung verwendeten Code kennen. Der Beitrag entschlüsselt das IR-Protokoll sowie die Codierung des IR Control Set (30344) und des aktuellen Nachfolgers Control Set (500881).

Teil 1: IR Control Set (30344)

Motivation

Bei der Übernahme einer gebrauchten fischertechnik-Sammlung bin ich in den Besitz eines fischertechnik-Infrarot-Empfängers aus dem *IR Control Set* (30344) gekommen, ohne passenden Sender. Diesen zu beschaffen erwies sich als schwierig: Bei fischerfriendsman ist er nicht lieferbar.

Abb. 1: fischertechnik IR Control Set (30344)

Es gibt eine anscheinend baugleiche Fernbedienung Rademacher 9490 für Rollladensteuerungen – diese kostet allerdings 46 €. Eine anlernbare Fernbedienung wäre die deutlich günstigere Lösung, allerdings muss diese zunächst angelernt werden: Ohne die originale Fernsteuerung musste ich mir also etwas anderes einfallen lassen.

Ermittlung der verwendeten Codes

Dazu habe ich mir mit einem Arduino Nano und ein paar Infrarot-LEDs, deren Strom über einen Transistor geschaltet wird, selbst einen Fernsteuerungs-Sender gebaut. Bekannt ist, dass fischertechnik den RC5 Code verwendet. Unbekannt waren die verwendete 5-bit-Adresse und die 6-bit-Kommandos. Diese musste ich durch Ausprobieren finden: In einem *Brute-Force*-Verfahren habe ich alle möglichen Kombinationen von Adressen und Kommandos getestet, bis der Empfänger jeweils die erwarteten Reaktionen gezeigt hat. Geholfen hat dabei auch die Steuerungsanleitung, die in der fischertechnik-Datenbank verfügbar ist [1]. Listing 1 zeigt die C-Funktionen, die ich zum Senden des IR-Codes benutzt habe.

Ergebnis

Der Empfänger arbeitet mit dem RC5-Code exakt nach Spezifikation, siehe z. B. die Beschreibungen auf opendcc und von Jörg Bredendiek [2, 3]. Mit einer 36-kHz-Trägerfrequenz konnte ich den Empfänger gut mit hoher Reichweite ansprechen. Ich habe die Kommandos jeweils doppelt gesendet und dabei das Toggle-Bit gewechselt.

```
volatile uint8_t timer_halfperiod = 0; //incremented with TCNT1 overflow
volatile uint8_t toggle = 0;

void init (void) // system clock 16 MHz is used (external quartz oscillator)
{       PORTD = _BV(PORTD2); // pull-up resistor on PD2 for IR receiver SFH5110
        DDRD = _BV(DDD3); // output PD3 for IR LED
        TIMSK0 = _BV(TOIE0);
        TCCR0B = _BV(CS02); //start TCNT0 with prescaler clock/256
// 16 MHz/1780/8 =  1124 Hz pwm frequency ( period of 890 µsec)
// timer 1 is used in fast pwm mode 14, WGM13:10 = "1110": ICR1 contains the top value for TCNT1
        TIMSK1 = _BV(TOIE1);
        TCCR1A = + _BV(WGM11); //CTC mode with ICR1 as top value
        TCCR1B = _BV(WGM13) + _BV(WGM12) + _BV(CS11); //TCNT1 with prescaler clock/8 (0.5 µs/step)
        ICR1 = 1780; //adapted to result in 890 µs for half the period of IR bits
// 16 MHz/8/55 =  36.36 kHz pwm frequency ( period of 27.5 µsec)
// timer 2 is used in fast pwm mode 7, WGM22:20 = "111": OCR2A contains the top value for TCNT2
        TCCR2A = _BV(WGM21) + _BV(WGM20); //_BV(COM2B1) is controlled in called function
        TCCR2B = _BV(WGM22) + _BV(CS21); //start TCNT2 with prescaler clock/8
        OCR2A = 55; //defines the period, i.e. the top value 16 MHz/8/56 = 36.36 kHz -> 27.5 µs
        OCR2B = 14; //defines the pulse length value: 0.5 µs * 14 = 7 µs
        EICRA = _BV(ISC01); // falling edge generates interrupt on INT0 (IR remote control)
        EIMSK = _BV(INT0); // enable INT0
        SMCR = _BV(SE); // idle mode is set up as sleep mode
        sei (); }
void send_0() {
        TCNT1 = 0;
        TCNT2 = 0;
        timer_halfperiod = 0; //incremented with TCNT1 overflow
        TCCR2A |= _BV(COM2B1); //activate OCR2B output to IR LEDs
        while (timer_halfperiod == 0) sleep_mode();
        TCCR2A &= ~(_BV(COM2B1)); //de-activate OCR2B output to IR LEDs
        TCNT1 = 0;
        timer_halfperiod = 0; //incremented with TCNT1 overflow
        while (timer_halfperiod == 0) sleep_mode();}
void send_1() {
        TCNT1 = 0;
        timer_halfperiod = 0; //incremented with TCNT1 overflow
        while (timer_halfperiod == 0) sleep_mode();
        TCNT1 = 0;
        TCNT2 = 0;
        timer_halfperiod = 0; //incremented with TCNT1 overflow
        TCCR2A |= _BV(COM2B1); //activate OCR2B output to IR LEDs
        while (timer_halfperiod == 0) sleep_mode();
        TCCR2A &= ~(_BV(COM2B1)); } //de-activate OCR2B output to IR LEDs
void send_RC5 (uint8_t address, uint8_t command) {
        uint8_t loops = 2;
        uint8_t pulse_count=14;
        uint8_t mask = 0b01000000; //mask MSB of inverted command bit C6
        if (mask & command) address |= 0x80; // add start bit and inverted command bit C6
        else address |= 0xC0; // add start bit and inverted command bit C6, toggle bit is 0
        do { //execute 2 times with pause and switch toggle bit
                mask = 0b10000000; //mask MSB of 8-bit address
                //send 8 address bits (two start bits, one toggle bit + five address bits)
                do {    if (address & mask) send_1();
                        else send_0();
                        mask = mask >> 1;
                        pulse_count --; } while (pulse_count > 6);
                mask = 0b00100000; //mask MSB of 6-bit command
                //send 6 command bits
                do {    if (command & mask) send_1();
                        else send_0();
                        mask = mask >> 1;
                        pulse_count --; } while (pulse_count);
                pulse_count = 14; //refresh for next transfer
                address |= 0x20; //set toggle bit
                loops --;
                lcd_wait_ms(89); } while (loops); }
```

Listing 1: C-Funktionen für den Arduino Nano zum Senden von RC5-Code

Die zeitliche Periode wiederholter Codes beträgt 114 ms, d. h. es gibt eine Pause von 89 ms zwischen den 14-bit-Übertragungen, die ca. 25 ms in Anspruch nehmen.

Ein weiteres, siebtes Command Bit, das in der später erweiterten Spezifikation anstelle des zweiten Startbits invertiert vorgesehen wurde (~C6), wird hier nicht genutzt. Daher sollte dieses Bit immer als zweites Startbit auf 1 gesetzt werden.

Der schwarze fischertechnik-IR-Empfänger (aus 30344) wird mit folgender Systemadresse angesprochen:

0x1B (hex) bzw. 27 (dec)

Taste	Command Code
M1 rückwärts	0x07
M1 vorwärts	0x08
M1 Leistung	0x03
M2 links	0x09
M2 rechts	0x0A
M2 Leistung	0x04
M3 Linkslauf	0x01
M3 Rechtslauf	0x02
M3 Leistung	0x05

Tab. 1: RC5-Fernsteuerungscodes für den fischertechnik-IR-Empfänger aus 30344

In Tab. 1 sind die Command-Codes der Tasten dargestellt. Diese sind entsprechend der Nummerierung der Tasten in der Steuerungsanleitung codiert. „M1 Leistung“ bedeutet, dass bei Motor 1 die Leistungsstufe umgeschaltet, d. h. zwischen langsam und schnell hin- und hergeschaltet werden soll.

Mit diesen Codes kann man den Empfänger nun z. B. mit einem Arduino-Board oder anderen Mikrocontrollern steuern.

Vermutlich benötigt der blaue Zweit-Empfänger, den es als Ergänzung gab, entweder eine andere Systemadresse oder zumindest andere Kommandos. Einen solchen habe ich aber nicht und kann es daher nicht überprüfen.

Teil 2: Control Set (500881)

Motivation

Mit der neuen Infrarot-Fernsteuerung von fischertechnik (500881) gab es 2008 einen entscheidenden Schritt zur Verbesserung der Steuerung von Fahrzeugen und Motoren allgemein, da es nun 16 Geschwindigkeitsstufen für beide Fahrtrichtungen gibt und eine Servo-Steuerung für Lenkungen mit einer Auflösung von 5 bit (oder 32 Stufen) möglich ist. Die Fernsteuerung lässt sich auch gut einsetzen, um Mikrocontroller-gesteuerte Fahrzeuge zu bewegen (siehe [4, 5]). Bei fischerfriendsman habe ich zu diesem Zweck einen Sender einzeln (ohne Empfänger) recht günstig erworben. Zur Implementierung eines passenden Empfängers gab es bereits in ftpedia 2/2016 einen interessanten Beitrag [5], in dem auch auf die Vorarbeit von Ad im wiki der ft:community verwiesen wurde [6]. Für die Implementierung eines Empfängers fehlen dort aber noch ein paar Detail-Informationen, die ich hier ergänze.

Ermittlung der verwendeten Codes

Zunächst habe ich mit Oszilloskop, Fototransistor und Pull-up-Widerstand an 9 Volt die Trägerfrequenz ermittelt. Der Sender nutzt 38 kHz als Trägerfrequenz für die Modulation. Bekannt ist bereits aus [5, 6], dass im RC-MM-Kodierverfahren 16 Pulse mit Puls-Distanz-Modulation übertragen werden, deren 15 Zeitabstände jeweils zwei Bits entsprechen, in Summe also 30 Bits.

Ich betrachte hier die Empfänger-Seite. Dort werden die gesendeten Pulse invertiert und es bietet sich daher an, die fallenden Flanken der empfangenen Pulse zu betrachten. Der Zeitabstand zwischen den fallenden Flanken benachbarter Pulse gibt jeweils den Wert der beiden codierten Bits an, siehe Tab. 2. Dort habe ich auch gleich die Wertebereiche für den Timer0 angegeben, die ich für die Erkennung mit einer Toleranz von ca. ±50 µs zulasse.

Zeitabstand	TCNT0 min	TCNT0 max	Wert
800 µs	47	53	0b00
900 µs	54	59	0b01
1000 µs	60	65	0b10
1100 µs	66	71	0b11

Tab. 2: Zuordnung des Zeitabstandes der fallenden Flanke der empfangenen Pulse zu den binär codierten Werten

Weiterhin ist bereits aus [6] die Struktur der übertragenen Information bekannt, allerdings fehlt dort die genaue Zuordnung der Bits und ihrer Werte zu den Auslenkungen der Joysticks. Ich habe mir daher selbst aus einem Arduino Nano Board mit Text-LCD und einem SFH5110-Empfänger am Interrupt-Pin INT0 einen Empfänger aufgebaut. Die Auswertung der Pulsabstände und Zuordnung der Codes erfolgte mit der in Listing 2 dargestellten Interrupt-Routine. Das *Most Signifikant Bit* (MSB) wird zuerst gesendet. Aufgrund der Code-Struktur bot sich eine Segmentierung der 30 Bits in acht 4-Bit-Nibble an. Das letzte Nibble wird von mir mit 0b11 hinten aufgefüllt. Die empfangenen 4-Bit-Nibble werden auf dem Text-LCD dargestellt.

Ergebnis

Daraus ergaben sich die in Tab. 3 zusammengefassten Informationen. Zu Beginn wird das Nibble[0] übertragen. Es hat immer den Wert 0b1000 = 0x8. Man könnte dies als Systemadresse nutzen und nur dann mit der Dekodierung fortfahren, wenn die Übereinstimmung der ersten vier Bits mit 0b1000 gegeben ist. Alternativ könnte man dies auch nutzen, um die verwendeten Puls-Zeiten im Empfänger auf 1,0 ms bzw. 0,8 ms zu kalibrieren.

Nibble[1] enthält den Status der beiden DIP-Schalter und der beiden Taster. Details zur Bedeutung der Einzelbits sind in Tab. 4 angegeben. Nibble[2] bis Nibble[5] enthalten die Auslenkung der Joysticks von der neutralen Mittelposition, welche jeweils mit 0 kodiert ist, nach oben oder unten (vertikal) bzw. rechts oder links (horizontal) bis zum Anschlag, der jeweils mit 0xF kodiert ist.

i	Inhalt des 4-Bit-Nibble[i]	Wertebereich
0	Kopf / Header	0x8 fix
1	DIP- und On/Off-Taster	0x0 bis 0xF
2	rechter Joystick: vertikale Auslenkung	0x0 bis 0xF
3	rechter Joystick: horizontale Auslenkung	0x0 bis 0xF
4	linker Joystick: vertikale Auslenkung	0x0 bis 0xF
5	linker Joystick: horizontale Auslenkung	0x0 bis 0xF
6	Auslenkungsrichtung: 0: unten / links 1: oben / rechts	0x0 bis 0xF
7	Schwanz (Tail) mit Parity Bit	0x3 oder 0xB

Tab. 3: Bedeutung der übertragenen Bits

Dieser Wert kann direkt als PWM-Wert zur Geschwindigkeitskontrolle eines Motors genutzt werden; insgesamt können so vier DC- oder Servo-Motoren gesteuert werden.

	Nibble[1]
Bit 3	DIP-Schalter 2: 0: "Off", 1: "On"
Bit 2	DIP-Schalter 1: 0: "Off", 1: "On"
Bit 1	Taster "Off": 0: nicht gedrückt 1: gedrückt
Bit 0	Taster "On": 0: nicht gedrückt 1: gedrückt

Tab. 4: Bits in Nibble[1]

Die Richtung der vier möglichen Joystick-Auslenkungen (oben versus unten bzw. rechts versus links) kann als Drehrichtung für die vier Motoren interpretiert werden. Diese ist in Nibble[6] bitweise kodiert, und zwar in der gleichen Reihenfolge für die vier Bits 3 bis 0, in der zuvor die Nibble [2] bis [5] übertragen wurden (Tab. 5 und Abb. 2).

	Nibble[6]
Bit 3	rechter Joystick: 0: unten, 1: oben
Bit 2	rechter Joystick: 0: links, 1: rechts
Bit 1	linker Joystick: 0: unten, 1: oben
Bit 0	linker Joystick: 0: links, 1: rechts

Tab. 5: Bits in Nibble[6]

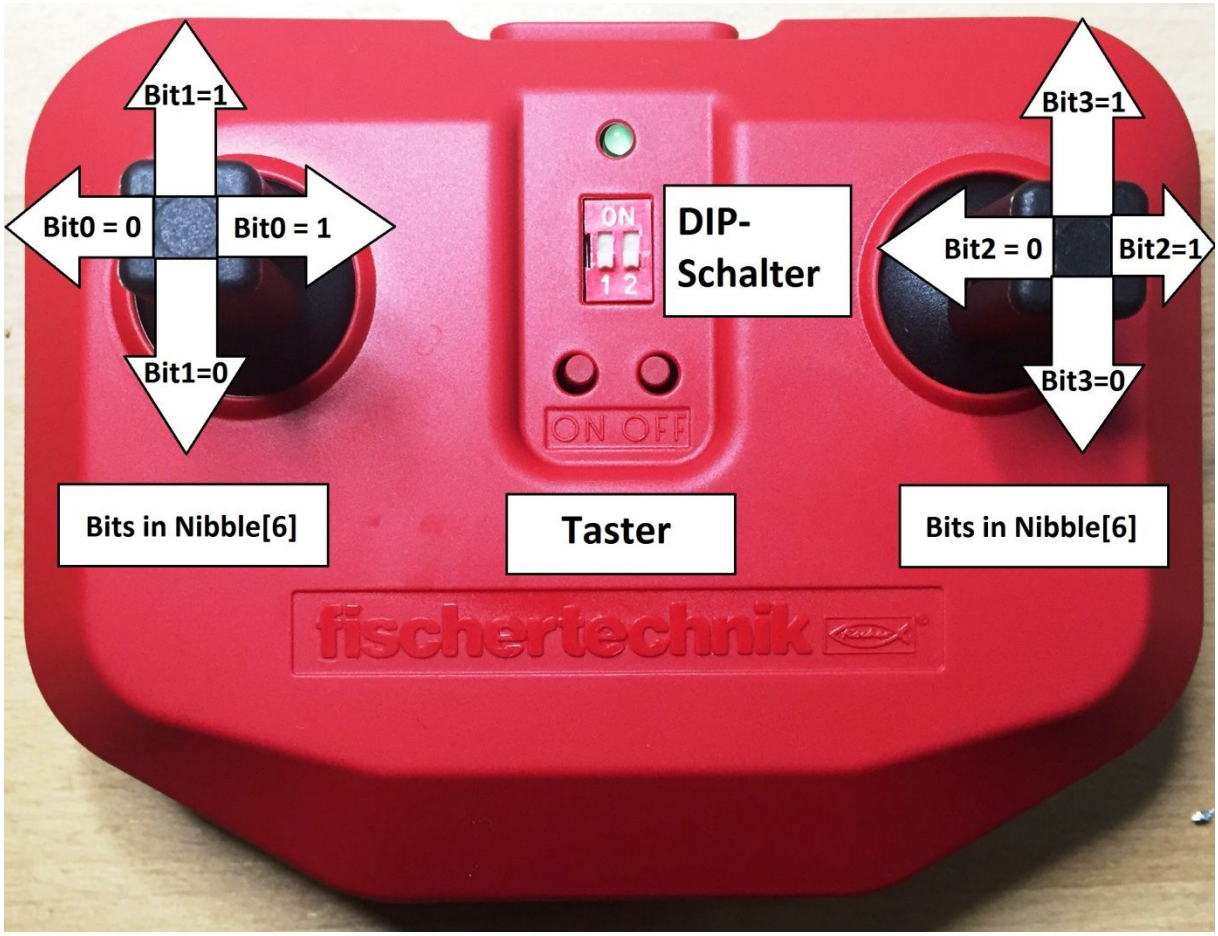


Abb. 2: IR-Fernsteuerungssender aus 500881
mit der Zuordnung der Richtungs-Bits in Nibble[6].

Nibble[7] enthält als MSB (Bit 3) das Parity-Bit, welches bei ungerader Anzahl der zuvor gesendeten Bits mit Wert 1 den Wert 1 annimmt, bei gerader Anzahl den Wert 0. Dies kann zur Verifikation der Übertragung genutzt werden oder auch, wie im Folgenden gezeigt, zur Fehlerkorrektur. Das Bit 2 ist immer 0 und die Bits 1 und 0 fülle ich absichtlich mit 0b11 auf zur Erkennung einer abgeschlossenen Übertragung im Hauptprogramm. Der Sender sendet seine 15 Pulse jeweils alle 120 ms, solange eine Taste betätigt wird oder ein Joystick außerhalb seiner Mittelstellung steht. Sonst ist Sendepause, um die Batterie zu schonen.

Fehlerkorrektur

Bei meinen Versuchen hat sich herausgestellt, dass es immer wieder auch zu Fehlern beim Empfang der Daten kommt. Zunächst habe ich die Übereinstimmung des Timings zwischen Sender und Empfänger überprüft. Dazu habe ich die Häufigkeit der TCNT0-Werte bei Pulsabständen im Bereich 750–850 µs ermittelt. Der Schwerpunkt lag bei TCNT0 = 50 (800µs), also genau passend. Der Schwingquarz auf dem Nano-Board macht sich für diese Anwendung bezahlt.

Ich konnte dann durch Messungen am Empfänger feststellen, dass öfter mal kurze Pausen zwischen den Pulsen vom Empfänger verschluckt wurden, so dass die Zeitdauer zwischen zwei Pulsflanken gelegentlich bei 1,6-1,9 ms lag, siehe Abb. 3.

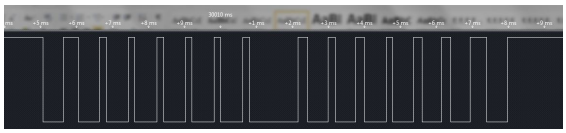


Abb. 3: Signal am Empfänger SFH5110 dargestellt im Logik-Analysator: nach dem achten Puls wurde eine Pause verschluckt.

Die Puls-Pausen liegen z. T unter 130 µs und damit bei weniger als fünf Perioden der Trägerfrequenz von 38 kHz. Anscheinend arbeitet der fischertechnik-Infrarot-Sender an der Spezifikationsgrenze für die gängigen IR-Empfänger.

Die Frage ist nun, ob die zugehörigen vier Bits noch ermittelt werden können. Da die Dauer der Pulse in etwa konstant ist (ca. 550-650 µs) und die Puls-Pausen für die Modulation variiert werden, ist es naheliegend anzunehmen, dass diese Fälle immer bei einem führenden Pulsabstand von 800 µs auftreten. Denn nur in diesem Fall wird die Dauer der Puls-Pause so kurz, dass sie nicht erkannt wird. Damit lassen sich die vier Bits entsprechend Tab. 6 ermitteln. Die Fehlerkorrekturen sind im Code in Listing 2 implementiert.

Zeitabstand	TCNT0 min	TCNT0 max	Wert
1600 µs	97	103	0b0000
1700 µs	104	109	0b0001
1800 µs	110	115	0b0010
1900 µs	116	121	0b0011

Tab. 6: Zuordnung des Zeitabstandes der fallenden Flanke der empfangenen Pulse zu den Werten bei nicht erkannter Puls-Pause

Nun hat mich noch die Häufigkeit dieser Fehler interessiert. Ich habe daher im Programm-Code für jeden Fehlerfall einen eigenen Fehlerzähler mitlaufen lassen, auch für die Parity-Fehler. Ich habe bei den Versuchen sowohl die Tasten gedrückt, als auch die Joysticks in allen möglichen Richtungen und mit allen möglichen Kombinationen bewegt.

Mit dem unteren Grenzwert für TCNT0 von 47 gab es ca. 10 % Fehler mit zu kurzen Pulsabständen. Daher habe ich den unteren Grenzwert auf 40 abgesenkt, womit diese Fehlerklasse verschwand, allerdings mit einem Anstieg der Parity-Fehler.

Tab. 7 zeigt das Ergebnis. Bei mehr als 6000 übertragenen 30-Bit-Worten liegt die größte Häufigkeit der Fehler bei Parity-Fehlern (20 % der Übertragungen). Diese Fehlerklasse wird nicht korrigiert. Damit sind ein Fünftel der Übertragungen nicht brauchbar.

nicht korrigierte Parity-Fehler	korrigierte Fehler: TCNT0 = 97 bis 103
20 %	10 %

Tab. 7: Häufigkeit der aufgetretenen Fehler bei ca. 6000 Übertragungen des 30-Bit Wortes (DIP-Schalter beide „Off“)

Die Fehler mit Zeitabständen der Pulsflanken bei 1600 µs treten in 10 % der Fälle auf und werden korrigiert. Die anderen Fehlerklassen sind vernachlässigbar. Ich hoffe, das sind genug Details für eure Implementierung eines Empfängers für 500881. Sicher fallen euch noch mehr Möglichkeiten für Fehlerkorrekturen ein. Ein Beispiel: Übergänge von Bits in Nibble[6] (Richtungswechsel der Joysticks) könnten nur bei niedrigen Werten für die Auslenkung des entsprechenden Joysticks zugelassen werden.

Quellen

[1] fischertechnik: *IR Control Set (30344)*. ft-Datenbank.

[2] opendcc: *RC5 IR-Codes*.

[3] sprut: *IR-Fernsteuerung – der RC-5 Code*.

[4] Dirk Wölffel: *Große Modelle mit nur einer fischertechnik-IR-Fernsteuerung ansteuern*. In dieser ft:pedia.

[5] Holtz, David: *Alternative Controller (2): Infrarot-Empfänger*. ft:pedia 2/2016, S. 60-67.

[6] Van der Weiden, Ad: *IR on RI (or IR Control Set and the Robo Interface)*. ft:community-Wiki, 2009.

```
volatile uint8_t timer_IR = 0; //incremented with TCNT0 overflow
volatile uint8_t pulse_count = 0xFF;
volatile uint8_t RC = 0xFF;
volatile uint8_t nibble[8] = { [0 ... 7] = 0 };
volatile uint8_t parity_counter = 0;
volatile uint8_t time_stamp_IR_low = 0;
volatile uint8_t time_stamp_IR_high = 0;

ISR (INT0_vect)
{     time_stamp_IR_low = TCNT0;
      time_stamp_IR_high = timer_IR;
      TCNT0 = 0;
      timer_IR = 0;
      if (time_stamp_IR_high > 5)//detect new start condition
      {     pulse_count = 0;
            RC = 0;
            parity_counter = 0; }
//Now the timing is synchronized to the relevant negative slope of //the inverted
IR pulses coding. After ca 750 - 1150 µs the next
//relevant edge should occur. positive edges in between are ignored
      else if (pulse_count < 15)
      {     if (time_stamp_IR_low > 121) pulse_count = 0xFF;
//longer than two pulses with 0b0011, error not fixed
            else if (time_stamp_IR_low > 115) //case 0b0011
            {     RC = 4*RC;
                  pulse_count ++;
                  if (!(pulse_count & 0x01)) //treat even case
                  {     nibble[(pulse_count/2) - 1] = RC;
                        RC = 0; }
                  RC = (4*RC) + 3;
                  pulse_count ++; }
            else if (time_stamp_IR_low > 109) //case 0b0010
            {     RC = 4*RC;
                  pulse_count ++;
                  if (!(pulse_count & 0x01)) //treat even case
                  {     nibble[(pulse_count/2) - 1] = RC;
                        RC = 0; }
                  RC = (4*RC) + 2;
                  pulse_count ++;
                  parity_counter++; }
            else if (time_stamp_IR_low > 103) //case 0b0001
            {     RC = 4*RC;
                  pulse_count ++;
                  if (!(pulse_count & 0x01)) //treat even case
                  {     nibble[(pulse_count/2) - 1] = RC;
                        RC = 0; }
                  RC = (4*RC) + 1;
                  pulse_count ++;
                  parity_counter++; }
            else if (time_stamp_IR_low > 96) //case 0b0000,fix error
            {     RC = 4*RC;
                  pulse_count ++;
                  if (!(pulse_count & 0x01)) //treat even case
                  {     nibble[(pulse_count/2) - 1] = RC;
                        RC = 0; }
                  RC = (4*RC);
                  pulse_count ++; }
```

```
            else if (time_stamp_IR_low > 71) pulse_count = 0xFF;
            else if (time_stamp_IR_low > 65 ) // case 0b11=3
            {     RC = (4*RC) + 3;
                  pulse_count ++; }
            else if (time_stamp_IR_low > 59 ) // case 0b10=2
            {     RC = (4*RC) + 2;
                  pulse_count ++;
                  parity_counter ++; }
            else if (time_stamp_IR_low > 53 ) // case 0b01=1
            {     RC = (4*RC) + 1;
                  pulse_count ++;
                  parity_counter ++; }
            else if (time_stamp_IR_low > 40 ) // case 0b00=0
            {     RC = 4*RC;
                  pulse_count ++; }
            else pulse_count = 0xFF; //error, pulse too short

            if (!(pulse_count & 0x01)) //do for even pulse_count
            {     nibble[(pulse_count/2) - 1] = RC;
                  RC = 0; }
            else if (pulse_count == 15)
//check parity, adjust last nibble
            {     if ((parity_counter&0x01)==((nibble[7]>>1)&0x01))
                        nibble[7] = (4*RC) + 3;
                  else nibble[7] = 0; }}}
                  //skip erroneous data (marker for main program)
```

Listing 2: Interrupt-Service-Routine zur Dekodierung der IR-Signale von 500881. Es wird dieselbe init()-Funktion genutzt wie beim RC5-Sender (siehe Listing 1).

Computing

RoboRISC: Visual Basic für den Robotics TXT

Andreas Gail

Es sollte auch für den Robotics TXT Controller eine Methode gefunden werden, um eine Programmierung über Visual Basic 2010 Express zu ermöglichen. Im folgenden Beitrag wird gezeigt, wie das mithilfe des durch fischertechnik bereitgestellten C++-Projekts möglich ist. Unter Visual Basic steht nun ein einfacher Befehlssatz bereit, „RoboRISC" genannt (RISC = Reduced Inststruction Set Computing). Das alles (bzw. sogar besser) war zwar auch schon mit dem alten Robo TX Controller machbar; Bei dem hier vorgestellten Projekt ist aber die Nutzung der Kamera und die Weiterverarbeitung der übertragenen Bilder von besonderem Interesse.

fischertechnik liefert die Grundlage

Über die Internetseite von fischertechnik wird das *TXT C Programming Expert Kit 4.1.6* (Stand 02.2016) zum Download bereitgestellt [1]. Dieser Download enthält ein C++-Projekt, welches sich unter Visual C++ 2010 Express öffnen lässt. Aus C++ heraus kann prinzipiell alles programmiert werden.

Ziel

C++ ist zwar verbreitet, allerdings muss man sich mit dieser Sprache gut auskennen, was nicht gerade trivial ist. Deshalb war das Ziel die Nutzbarkeit unter Visual Basic 2010 Express (oder höher) zu ermöglichen. BASIC steht für „Beginner's All-purpose Symbolic Instruction Code". Sicherlich geht vielen BASIC deutlich leichter von der Hand als C++.

Konzept-Idee

Zunächst sollten wichtige Befehle im bestehenden C++-Projekt ergänzt werden, was dann zusammen als DLL zu kompilieren wäre. Diese DLL wiederum soll dann in ein VB-Projekt eingebunden werden. Im VB-Projekt wiederum können letztendlich alle Funktionalitäten programmiert werden, einmal die Windows-basierte Bedien- und Anzeigeoberfläche sowie die eigentliche Automatisierung des fischertechnik Modells unter Ansteuerung des Robotics TXTs.

Optionen und Einschränkungen von RoboRISC

Möglich ist das digitale Auslesen der Eingänge des Robotics TXT I1…8 sowie die digitale Ansteuerung der Ausgänge O1…8 bzw. M1…4. Analoge Funktionalitäten (Analogeingänge und verschieden stark angesteuerte Ausgänge) sind bislang nicht implementiert.

Wegen der unterschiedlichen Verarbeitungsgeschwindigkeiten der beiden unabhängigen Systeme (PC und Robotics TXT) dürfen aufeinander folgende Befehle nicht zu schnell aufeinander folgen. Dieses Problem wird systematisch kanalisiert unter Nutzung des *JobDispatchers* unter Visual Basic (siehe weiter unten). Damit werden Kommunikationsstörungen vermieden.

Gleich zu Beginn soll die begrenzte Ausführungsgeschwindigkeit von RoboRISC nicht verschwiegen werden.

Für den interessierten Leser sei erwähnt, dass die nutzbaren Funktionen als *native*

code bereitgestellt werden, nicht als *managed code*. Das ist bei der späteren Einbindung unter Visual Basic von Bedeutung.

Implementierung in Visual Basic

Nach dem Kompilieren der DLL müssen die darin enthaltenen Funktionen für VB bereitgestellt werden. Hierzu muss man wissen, wie die Funktionen genau heißen, bzw. wie sie aufzurufen sind. Das kann über den *Microsoft COFF/PE Dumper* herausgefunden werden. Hierzu müssen folgende Schritte ausgeführt werden:

- Folgendes Kommando als Direktbefehl im Dialogfeld *Start, Ausführen…* gemäß Abb. 1 eingeben (gezeigt am Beispiel von Windows XP mit Service Pack 3):

```
%comspec% /k
""C:\Programme\Microsoft Visual
Studio 10.0\VC\vcvarsall.bat"" x86
```

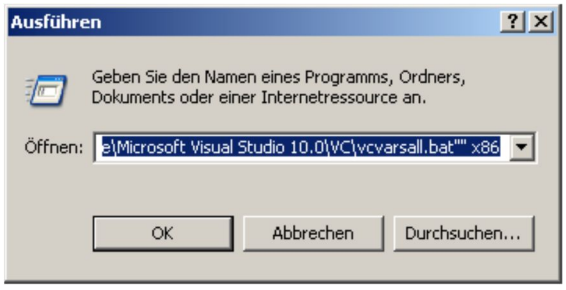


Abb. 1: Öffnen des Direktfensters

Nach der Befehlsausführung öffnet sich folgendes Eingabefenster:

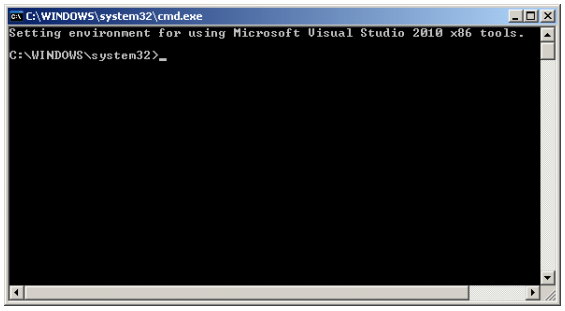


Abb. 2: Direktfenster

- Das Analyseprogramm wird ausgeführt, indem folgende Kommandozeile eingeben wird (\...\ enthält dabei den vollständigen Pfad, und der Zeilenumbruch ist nur der Spaltenbreite geschuldet – bitte alles in eine Zeile eingeben):

```
"C:\...\dumpbin.exe" /exports
"C:\...\RoboTXT_v01.dll"
```

- Im Direktfenster gemäß Abb. 2 wird das Analysenergebnis ausgegeben. Somit stehen folgende Funktionen bereit:

```
?GetDigitalAll@TXT@@QAEIHHHH@Z
?GetPicture@TXT@@QAE_NHHHH@Z
?SetDigitalAll@TXT@@QAEIHHHHHHHH@Z
```

Anschließend muss die Funktion unter Visual Basic deklariert werden (siehe Abb. 3). Was die Zeichen hinter dem Funktionsnamen bedeuten, ist dabei unerheblich.

Bereitgestellte Funktionen der Klassenbibliothek TXT und deren Nutzung

TXT.GetPicture(IPa, IPb, IPc, IPd)

- IPa = UINT16: IP Adresse 1. Zahl
- IPb = UINT16: IP Adresse 2. Zahl
- IPc = UINT16: IP Adresse 3. Zahl
- IPd = UINT16: IP Adresse 4. Zahl
- Rückgabe: Boolean: True, wenn erfolgreich

Das übertragene Bild wird im Verzeichnis der aktuellen Anwendung gespeichert. Es hat den Dateinamen *RoboTxtPic.JPG*.

Beispiel: vollständige IP-Adresse des Controllers = 192.168.7.2

- IPa = 192
- IPb = 168
- IPc = 7
- IPd = 2

TXT.GetDigitalAll(IPa, IPb, IPc; IPd)

- IPa = UINT16: IP Adresse 1. Zahl
- IPb = UINT16: IP Adresse 2. Zahl
- IPc = UINT16: IP Adresse 3. Zahl
- IPd = UINT16: IP Adresse 4. Zahl

- Rückgabe: UINT16: Der Rückgabewert enthält alle acht Eingangszustände (I1...8)

Beispiel: Auswertung des zurückgegebenen Funktionswertes

- I1 = 1
- I2 = 2
- I3 = 4
- I4 = 8
- I5 = 16
- I6 = 32
- I7 = 64
- I8 = 128

Wenn der Rückgabewert z. B. 5 ist, dann waren zum Abfragezeitpunkt die Eingänge I1 und I3 geschaltet.

TXT.SetDigitalAll(Output, maxTime, Input1, Input2, IPa, IPb, IPc, IPd)

- Output = UINT16: Codierung der Einschaltung der Ausgänge O1...8
- Input1 = UINT16: Warten auf Änderung von Eingang I1...8
- Input2 = UINT16: Warten auf Änderung von Eingang I1...8
- IPa = UINT16: IP Adresse 1. Zahl
- IPb = UINT16: IP Adresse 2. Zahl
- IPc = UINT16: IP Adresse 3. Zahl
- IPd = UINT16: IP Adresse 4. Zahl
- Rückgabe: UINT16: Laufzeit der Funktion in s

Gemäß den Output-Parametern werden die Ausgänge gleichzeitig geschaltet. Die Ausgangsleistung ist einheitlich maximal. Wenn z. B. ein Motor M1 betrieben werden soll, so dreht sich dieser, wenn O1 = 1 und O2 = 0. Bei der umgekehrten Einstellung O1 = 0 und O2 = 1 dreht sich der Motor dann in die andere Richtung.

Die Wiederabschaltung erfolgt automatisch entsprechend dem Wert des Parameters *maxTime*. Dieser Vorgang kann abgebrochen werden, wenn sich ein oder zwei digitale Eingangswerte ändern.

Beispiel: Output Parameter

- O1 = 1
- O2 = 2
- O3 = 4
- O4 = 8
- O5 = 16
- O6 = 32
- O7 = 64
- O8 = 128

Wenn z. B. die Ausgänge O1 und O3 gleichzeitig eingeschaltet werden sollen, wäre Output = 5 festzulegen.

Der JobDispatcher

Wie bereits oben erwähnt, sind der PC und der Robotics TXT zwei autarke Systeme, die per USB-Kabel miteinander verbunden sind. Beide Systeme müssen gleichzeitig bereit sein, um eine Kommunikation zu ermöglichen. Aus diesem Grund wurde experimentell eine gewisse Wartezeit ermittelt, nach dem der nächste Kommunikationsvorgang sicher möglich ist.

Die Einhaltung der Wartezeiten wird dabei vom *JobDispatcher* kontrolliert. Damit das möglich wird, müssen alle erforderlichen Vorgänge in die *JobList* aufgenommen werden, aus der diese dann sequentiell abgearbeitet und schließlich wieder entfernt werden.

Ein diesbezüglich kritischer Vorgang ist die Übermittlung von Bildern. Diese werden erst einige Zeit nach dem Absenden der Übermittlungsanfrage vom Robotics TXT zum PC übertragen. Neben immer wieder auftretenden Übermittlungsfehlern, die abgefangen werden müssen, darf auf diese Bilder vom PC erst dann für eine Weiterverarbeitung zugegriffen werden, wenn die Übertragung erfolgreich abgeschlossen ist. Auch das wird über den JobDispatcher sichergestellt.

Verwendung des Beispielprogramms

Im bereitgestellten Beispielprojekt können drei prinzipielle Funktionalitäten demonstriert werden (siehe Abb. 4 und [2]). Hierbei ist zu beachten, dass sich die Datei *RoboTXT_v01.dll* im Verzeichnis *C:\Programme\RoboTXT_LIB* befinden muss, oder alternativ in der Datei *RoboTXT.vb* der passende Pfad einzustellen ist.

a) Verbindungsaufbau zwischen RoboTXT und PC aufbauen

Der Verbindungsaufbau ist denkbar einfach; im Grunde reicht die Betätigung der Befehlsschalfläche *check*. Wenn eine Verbindung besteht, wird es angezeigt.

b) Digitale Eingänge auslesen

Der aktuelle digitale Zustand der Eingänge kann mithilfe der Befehlsschaltfläche *I1...8 digital* ausgelesen werden. Fotowiderstände und Fototransistoren können dabei auch als digitale Signalgeber eingesetzt werden. Der aktuelle Zustand wird über die entsprechenden Kontrollkästchen angezeigt.

c) Digitale Ausgänge einstellen

Die Auswahl digitaler Ausgänge, die nachfolgend angesteuert werden sollen, kann über die zugehörigen Kontrollkästchen definiert werden. Für Motoren gibt es die Besonderheit, dass (wie oben bereits erwähnt) gleichzeitig zwei Ausgänge definiert werden müssen.

Weiterhin muss eine Zeit festgelegt werden, für die die gewünschte Ansteuerung vorgenommen werden soll. Schließlich können maximal zwei Eingänge definiert werden, die zu einem Abbruch der Ausgangsansteuerung führen. Hierbei wird betrachtet, ob es zur Laufzeit zu einer Änderung der Eingangsbeschaltung kommt.

d) Kamerabild anfordern

Über die Befehlsschaltfläche *Camera* kann ein aktuelles Kamerabild angefordert werden.

Ausblick (nah)

Auf den ersten Blick mag der eine oder andere das hier vorgestellte Verfahren im Vergleich zu anderen Programmiermöglichkeiten des Robotics TXT als recht eingeschränkt empfinden. Allerdings wird in einem weiteren Beitrag dieser Ausgabe der ft:pedia gezeigt, dass durchaus recht komplexe Automatisierungsaufgaben unter Nutzung von RoboRISC und Windows-Funktionalitäten machbar sind [3].

Ausblick (fern)

Vielleicht lässt sich über die ft:pedia oder die ft-Community eine Programmbibliothek (DLL) erstellen, mit der alle bekannten Robotics TXT Controller-Funktionen in Microsoft .NET managed code bereitgestellt werden können. Im Idealfall wäre alles auch gleich kompatibel mit dem Robo TX Controller.

Quellen

[1] fischertechnik: *ROBOTICS TXT Controller C-Programming Kit Firmware Version 4.1.6*. Fischerwerke, 2015.

[2] Gail, Andreas: *RoboTXTC link to VisualBasic2010 digital I/O camera access and bar code reader*. Visual Basic 2010-Beispiel-Programm.

[3] Gail, Andreas: *Hochregallager mit Kamera-Strichcodeleser, Microsoft Visual Basic 2010 und RoboRISC*. In dieser ft:pedia.

```
<DllImport(dllpath, EntryPoint:="?GetDigitalAll@TXT@@QAEIHHHH@Z", _
CallingConvention:=CallingConvention.StdCall)> _
<DllImport(dllpath, EntryPoint:="?GetDigitalAll@TXT@@QAEIHHHH@Z", _
CallingConvention:=CallingConvention.StdCall)> _
Public Shared Function GetDigitalAll(ByVal IPa as integer, _
ByVal IPb as integer, ByVal IPc as integer, ByVal IPd as integer) As UInteger
End Function

<DllImport(dllpath, EntryPoint:="?GetPicture@TXT@@QAE_NHHHH@Z", _
CallingConvention:=CallingConvention.StdCall)> _
Public Shared Function GetPicture(ByVal IPa as integer, _
ByVal IPb as integer, ByVal IPc as integer, ByVal IPd as integer) As Boolean
End Function

<DllImport(dllpath, EntryPoint:="?SetDigitalAll@TXT@@QAEIHHHHHHHH@Z", _
CallingConvention:=CallingConvention.StdCall)> _
Public Shared Function SetDigitalAll(ByVal Output as integer, _
ByVal maxTime as integer, ByVal Input1 as integer, ByVal Input2 as integer, _
ByVal IPa as integer, ByVal IPb as integer, ByVal IPc as integer, _
ByVal IPd as integer) As UInteger
End Function
```

Abb. 3: Deklarationen der Funktionen in „RoboTXT_v01.dll“

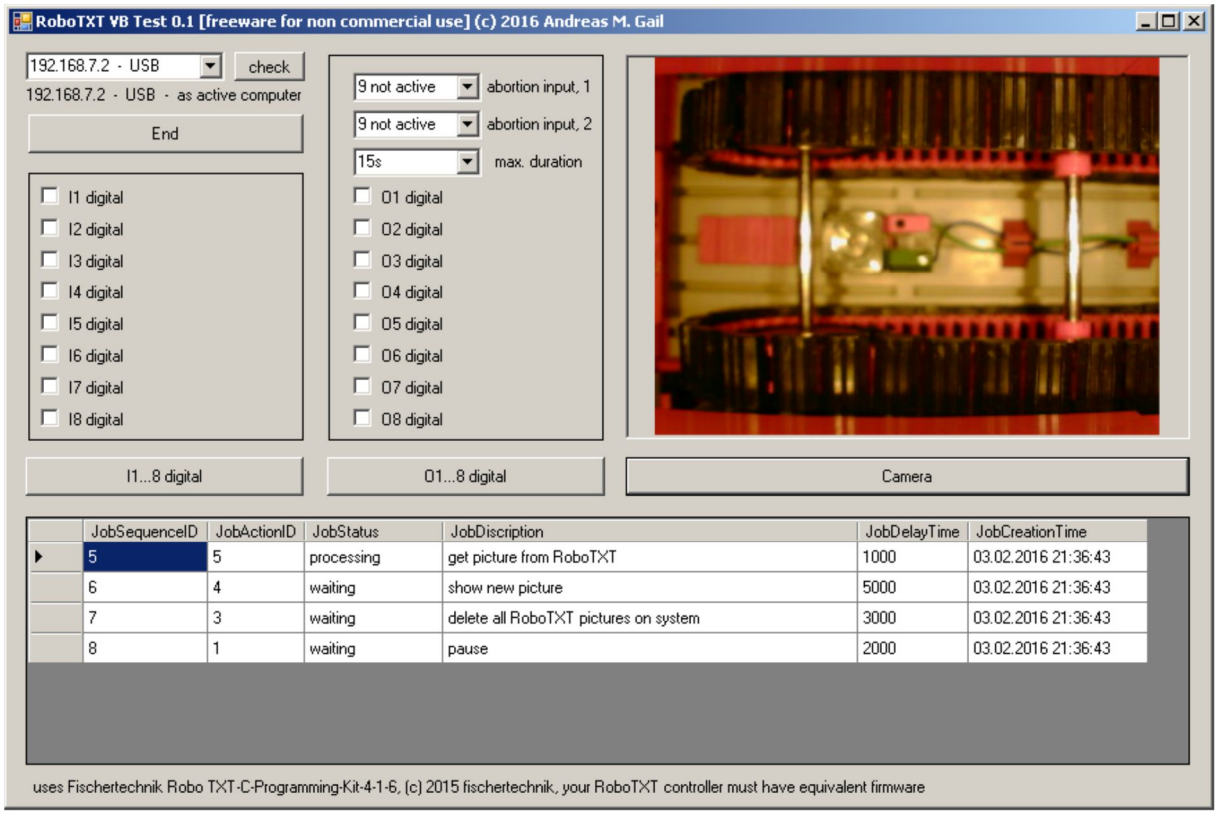


Abb. 4: Visual Basic 2010-Beispielprogramm zu Nutzung der „RoboTXT_v01.dll“

Computing

Hochregallager mit Kamera-Strichcodeleser, Microsoft Visual Basic 2010 und RoboRISC

Andreas Gail

fischertechnik-Modelle von Hochregal-Lagern gibt es viele im Internet. So wurde z. B. auch gezeigt, wie unterschiedlich gefärbte Teile automatisch erkannt und eingelagert werden. Inzwischen bietet fischertechnik eine Kamera (für die Nutzung mit einem Robotics TXT Controller) an, mit der es möglich ist, Strichcodes (bar codes) zu lesen. Das zeigt der folgende Beitrag, in dem Microsoft Visual Basic 2010 und RoboRISC (siehe den vorausgehenden Beitrag in dieser Ausgabe der ft:pedia) eingesetzt werden.

Anforderungen an ein vollautomatisches Hochregal-Lager

Das Modell muss Folgendes leisten:

- Auf einem Förderband abgestellte Ware soll erkannt und automatisch eingelagert werden.
- Ein- und Auslagerung sollen vollautomatisch verlaufen.
- Um welche Ware es sich bei der Einlagerung handelt, soll mithilfe der fischertechnik-Kamera automatisch erfasst und in einer Warenbestandsliste eingetragen werden. Um das zu ermöglichen, ist die Ware mit einem Strichcode versehen. In der Warenbestandsliste sollen die Klartextbezeichnung der Ware und der Einlagerungszeitpunkt eingetragen werden.
- Einlagerungen sollen nur vorgenommen werden, wenn auch noch ein freier Lagerplatz vorhanden ist.
- Die Vergabe der Lagerplätze soll so vorgenommen werden, dass die benötigten Zeiten für den Einlagerprozess so klein wie möglich sind.
- Die Warenbestandsliste soll nach verschiedenen Kriterien sortiert werden können.
- Zur Auslagerung soll ein Doppelklick auf einen Eintrag in der Warenbestandsliste ausreichen.
- Die ausgelagerte Ware soll aus der Warenbestandsliste gelöscht werden.

Einschränkungen der Strichcode-Erkennung

Folgende Randbedingungen müssen eingehalten werden, damit der Strichcode erfolgreich erkannt werden kann:

- Nur Verwendung von Codes vom Typ Code 39.

- Nur vierstellige Codes (4 Ziffern).
- Rechts und links vom Code muss das erfasste Kamerabild weiß sein.
- Diagonale Codeerfassung ist nicht möglich.
- Die Kamera muss den Code scharf erfassen können (Objektiveinstellung) und möglichst bildfüllend.
- Ausreichende, möglichst gleichverteilte (diffuse) Beleuchtung, hierzu eignen sich weiße LEDs.

Wenn obige Einschränkungen berücksichtigt werden, ist eine horizontale oder vertikale Code-Erfassung möglich, auch kopfstehend.

Kamera Strichcode-Leser

Die Besonderheit dieses fischertechnik-Modells ist sicherlich, dass ein Strichcode mit einer Kamera erfasst werden kann und eine Auswertung des Bildes in Visual Basic stattfindet. Dabei werden nachfolgende Schritte automatisch ausgeführt:

a) Anzeige des übertragenen Kamerabildes in einem Anzeigefeld (PictureBox). Hierbei wird eine Konvertierung aus dem komprimierten Bildformat (.jpg) nach Bitmap (.bmp) vorgenommen.

b) Die horizontale und vertikale Anzahl der Bildpunkte des bereitgestellten Kamerabildes wird analysiert.

c) Bei einem Stichcode ist ein Streifen entweder schwarz oder weiß. Farben oder Graustufen im aufgenommenen Bild stören bei der späteren Auswertung. Ausgegangen wird von einem Bild gemäß Abb. 1.

Wenn man sich das Bild stark (achtfach) vergrößert ansieht (Abb. 2), sieht man, dass die Streifen des Strichcodes in Wirklichkeit gar nicht so scharf abgebildet wurden, wie man vielleicht zuerst vermutet hatte.

Abb. 1: aufgenommenes Bild, unverändert

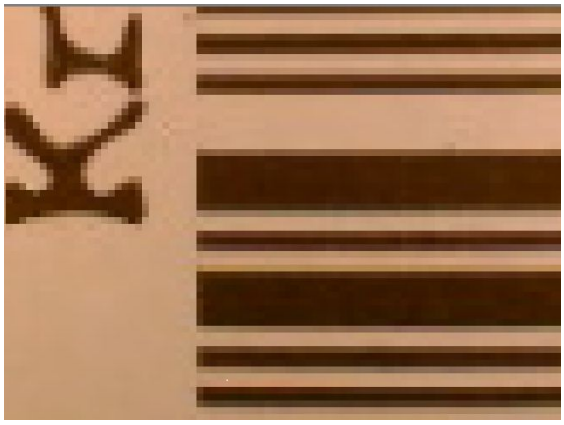


Abb. 2: aufgenommenes Bild, vergrößert auf 800%

Aufgrund der vorhandenen Unschärfe wird das Bild aus Abb. 1 in ein reines Schwarz-Weiß-Bild umgerechnet, bei dem es keine Farben und Graustufen mehr gibt. Hierzu muss man wissen, dass ein Farbbild mit 24 Bit Farbtiefe aus den drei Einzelfarben rot, grün und blau zusammengemischt wird.

Für jede dieser Farben stehen 8 Bit zur Verfügung, die 256 verschiedene Intensitätsstufen ermöglichen. Je kleiner der individuelle Farbwert ist, desto dunkler ist dieser. Bezüglich rot z. B. bedeutet 0 gar kein rot (d. h. keine rote Helligkeit) und 255 maximal helles Rot. 127 steht demnach für rot mit mittlerer Helligkeit. Dasselbe gilt entsprechend auch für grün und blau.

Wenn man also die Helligkeit eines Bildpunktes wissen möchte, so muss man für alle drei Farben den Helligkeitswert bestimmen und dann den Mittelwert bilden. Dieser

Mittelwert entscheidet dann zusammen mit der Einstellung gemäß Abb. 3, ob ein Bildpunkt schwarz oder weiß interpretiert werden soll.

Abb. 3: Schwarz/weiß-Schwellenwert-Einstellung

Die Einstellung, ab welcher Helligkeit ein Bildpunkt schwarz oder weiß werden soll, wird auch als Schwellenwert bezeichnet. Wird das Bild in Abb. 1 nach dem obigen Verfahren umgerechnet, so ergibt sich folgendes Bild:

Abb. 4: Bild, mit Schwellenwert in reines schwarz-weiß umgerechnet

d) Nun werden Streifenmuster aus dem Schwarz-Weiß-Bild ausgelesen. Das erfolgt entlang der roten und grünen Linien, wie in Abb. 5 gezeigt. Der Betrachter erkennt sofort, dass hierbei nur bei vier roten Linien die Aussicht auf Erfolg besteht, den Strichcode richtig auszulesen. Der Computer weiß das jedoch nicht und nimmt zunächst stur alle 20 Streifenmuster auf (rote und grüne). Damit die Linien sinnvoll über das Bild verteilt sind und bekannt ist, wie viele Punkte ausgelesen werden müssen, wurde zuvor gemäß b) die horizontale und vertikale Anzahl der Bildpunkte analysiert. Die roten und grünen Linien sind hier nur zur Erklärung dargestellt, bei einer wirklichen Auswertung würde das stören.

Abb. 5: Auswertung der Streifenmuster entlang der roten und grünen Linien

e) Das Auslesen entlang einer roten oder grünen Linie erfolgt Punkt für Punkt nacheinander. In einem Datenfeld (array) wird dieses Leseergebnis in komprimierter Form gespeichert. Würden z. B. zu Beginn fünf weiße Punkte hintereinander gefunden und der sechste Punkt wäre schwarz, dann würde in dem ersten Datenfeldeintrag eine 5 gespeichert werden. Wären dann der sechste bis achte Punkt weiß, wäre der nächste Datenfeldeintrag eine 3, usw. Sind alle roten und grünen Streifen gemäß Abb. 5 ausgelesen, hat man schließlich 20 derartige Zahlenfolgen.

f) Wie eingangs festgelegt, dürfen sich vor und nach der Strichcodeabbildung nur weiße Bildpunkte befinden. Das wird bei der Auswertung der Zahlenfolge im Datenfeld berücksichtigt. Aus den verbleibenden Datenfeldeinträgen wird die Summe gebildet und man erhält die Anzahl an Bildpunkten, die zum Strichcode gehören (könnten). Um die Datenfelder richtig auswerten zu können, sind die Grundlagen der Streifencodes gemäß einer früheren Ausgabe der ft:pedia von Bedeutung [3]. Für einen Strichcode mit vier Ziffern ergibt sich Folgendes:

- Ein dicker Streifen ist so breit wie zwei dünne.

- Eine Ziffer hat immer drei dicke und sechs dünne Streifen.
- Eine Ziffer hat eine Gesamtbreite von zwölf dünnen Streifen.
- Zwischen je zwei Ziffern liegt immer ein dünner Streifen.
- Wenn vier Ziffern nebeneinanderstehen, sind diese durch insgesamt drei dünne Streifen voneinander getrennt.
- Alle obigen Aussagen zusammen ergeben für vier Ziffern eine Gesamtbreite von 51 dünnen Streifen.

Die oben gebildete Anzahl an Bildpunkten, die zum Strichcode gehören, wird nun durch 51 geteilt. Damit erhält man die Anzahl an Bildpunkten, die einen dünnen Streifen darstellen. Die doppelte Bildpunktanzahl entspricht dann einem dicken Streifen.

g) Zusammen mit dem Wissen, wieviel Bildpunkte ein dicker bzw. ein dünner Streifen hat, können nun die Zahlenfolgen der Datenfelder gemäß e) analysiert werden. Daraus ergibt sich für eine Ziffer eine neue Zahlenfolge, bestehend aus „1“ (dünner Streifen) und „2“ (dicker Streifen). Es ergibt sich somit z. B. für die Ziffer 7 Folgendes:

- „111211212“ = 7, vorwärts gelesen
- „212112111“ = 7, rückwärts gelesen

Diese Zahlenfolgen, bestehend aus „1“ und „2“, werden nun durch Mustervergleich in einzelne Ziffern umgerechnet. Schließlich ergeben sich Zahlen mit jeweils vier Ziffern.

h) Die Vorgänge e) bis g) werden schließlich auf alle 20 Streifenmuster (rote und grüne) gemäß d) angewendet.

i) Die Ergebnisse aus h) werden schließlich betrachtet. Werden zwei oder mehr Streifenmuster als gültige Strichcodes interpretiert, so dürfen sich diese nicht

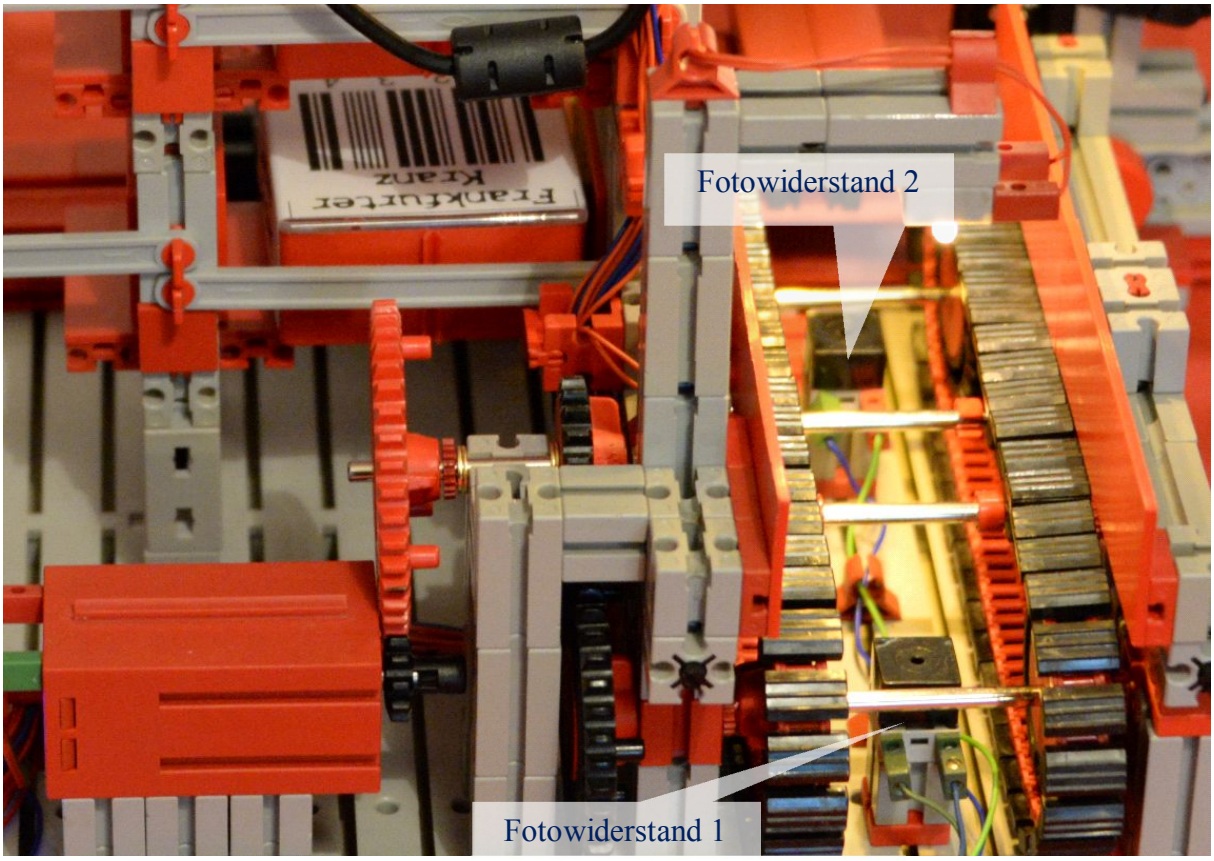


Abb. 6: Förderband mit Fotowiderständen

voneinander unterscheiden. Falls doch, wird die Fehlermeldung „bar code with errors detected“ als Ergebnis ausgegeben. Wird kein Streifenmuster als gültiger Strichcode erkannt, lautet die Fehlermeldung „no bar code detected“. Nur eindeutige Leseergebnisse werden als Zahlenwert ausgegeben.

Beim nachfolgend beschriebenen Modell ist es trotz hunderten von Durchgängen kein einziges Mal dazu gekommen, dass ein Strichcode als solcher erkannt, jedoch eine falsch gelesene Zahl ausgegeben wurde. Das beschriebene Verfahren wird somit als recht zuverlässig angesehen.

Das Modell

Das Modell selbst ist vergleichsweise einfach aufgebaut. Soll ein Gebinde eingelagert werden, muss es zunächst auf das Förderband aufgebracht werden. Die Einlagerung wird nur begonnen, wenn noch mindestens ein Lagerplatz frei ist. Im Bereich des Förderbandes sind gemäß Abb. 6 zwei Fotowiderstände eingebaut, mit denen die Lagergebinde auf dem Förderband erkannt werden können.

Wird also zwischen Fotowiderstand 1 und der darüber befindlichen weißen LED ein Lagergebinde eingebracht, so wird es erkannt. Das Lagergebinde wird darauf hin auf dem Förderband so weit transportiert, bis es sich über Fotowiderstand 2 befindet. Um das Lagergebinde sicher zu erkennen, muss eine passende Störlichtkappe über dem Fotowiderstand verwendet werden. An der Position des Fotowiderstandes 2 wird nun ein Kamerabild vom Gebindeetikett aufgenommen und ausgewertet. Im Modell sind weiße LEDs verbaut, weil lange Betriebszeiten erwartet sowie eine örtlich und zeitlich konstante Beleuchtung benötigt wurden. Wie bereits oben beschrieben, wird die erkannte Ware in der Warenbestandsliste eingetragen. Um eine Klartextbeschreibung in dieser Liste ergänzen zu können ist die Datei *StockList.txt* erforderlich, in der der Strichcode und die gesuchte Klartextbeschreibung vorhanden sind:

- 1234-Frankfurter Kranz
- 5678-Gruene Sosse
- 9059-Handkaes mit Musik
- 9088-Frankfurter Wuerstchen
- 1244-Bethmaennchen
- 7983-Rindswuerste

Die Liste kann natürlich beliebig ergänzt werden.

In der Warenbestandsliste sind auch alle bislang verwendeten Lagerpositionen vermerkt. Soll die nächstgelegene freie Lagerposition ermittelt werden, wird in einer weiteren (fest im Programm hinterlegten) Aufzählung nachgesehen. Die Lagerbelegung erfolgt gemäß folgendem Schema:

	A	B	C
4	**7**	**10**	**12**
3	**4**	**8**	**11**
2	**2**	**5**	**9**
1	**1**	**3**	**6**

Abb. 6: Schema der Lagerbelegung

Die möglichen Lagerpositionen sind grau hinterlegt. Die Reihenfolge der Belegung ist mit den roten Zahlen gekennzeichnet. Somit wird die Lagerposition A1 zuerst belegt, dann folgt A2, B1, A3, usw. Wenn die nächste freie Lagerposition bestimmt wurde, beginnt der Transport des Lagergebindes dorthin. Der Laderoboter übernimmt dabei das Lagergebinde vom Förderband. Er kann das Lagergebinde gemäß Abb. 12 in den Richtungen x, y und z transportieren und schließlich am festgelegten Lagerort einbringen. Aufgrund der minimalistischen Konstruktion des Laderoboters kann dieser nur dann in x-Richtung verfahren werden, wenn die Bewegungseinheit der y-Richtung an der tiefsten Stelle positioniert ist. Die Bildschirmdarstellung eines gefüllten Lagers ist in Abb. 13

gezeigt. Zur Auslagerung genügt ein Doppelklick auf einen Eintrag in der Warenbestandsliste. Vor Beginn der Auslagerung muss diese wie in Abb. 7 gezeigt bestätigt werden:

Abb. 7: Dialogbox zur Auslagerung eines Lagergebindes

Etikettierung der Lagergebinde

Abb. 8: Beklebtes Lagergebinde

An die Etikettierung der Lagergebinde werden auch einige Anforderungen gestellt.

- Der Strichcode sollte gut lesbar und so groß wie möglich sein, zumindest hinsichtlich der Breite. So können schmale und breite Streifen mit maximaler Zuverlässigkeit unterschieden werden.
- Eine Klartextkennzeichnung ist stets sinnvoll für Kontrollzwecke durch Menschen.
- Die Kästchen sollen durch die Verklebung sicher verschlossen gehalten werden.
- Rechts und links vom Strichcode soll kein Aufdruck vorhanden sein, weil das eine wichtige Voraussetzung für ein fehlerfreies Auslesen ist.
- Die Verklebung soll reversibel sein, d. h. zu einem späteren Zeitpunkt soll sie rückstandsfrei wieder entfernt werden können.

Ein paar Beispiele zur Etikettierung sind in Abb. 15 am Ende dieses Artikels zu finden. Vor dem Ausdruck ist zu beachten, dass die Abbildung derart vergrößert werden muss, dass ein Etikett die Größe von 5,5 cm · 10,8 cm hat.

Als Klebstoff eignet sich ein wasserlöslicher Klebestift, z. B. ein Pritt-Stift (Abb. 9). Der Klebstoff sollte zunächst großzügig, aber gleichmäßig auf die Rückseite des Etiketts aufgetragen und danach das Etikett umgehend auf das Kunststoffkästchen faltenfrei aufgebracht werden.

Abb. 9: Beispiel für einen wasserlöslichen Klebestift zur reversiblen Gebindeetikettierung

Enthält ein Klebstoff organische Lösemittel, so besteht die Gefahr, dass bei der Verklebung der Kunststoff angelöst wird. Derartige Klebstoffe sind ungeeignet.

Ausblick

Das Dekodieren von Strichcodes ist im Grunde nicht besonders kompliziert, wenn das Grundprinzip einmal richtig verstanden wurde. Deshalb erscheint es mir vergleichsweise einfach machbar, wenn in die fischertechnik Programmierumgebung ROBO Pro zukünftig ein Sensorfeld eingebaut würde, mit dem Strichcodes gelesen werden könnten.

Quellen

[1] fischertechnik: *ROBOTICS TXT Controller C-Programming Kit Firmware Version 4.1.6*. Fischerwerke, 2015.

[2] Gail, Andreas: *RoboTXTC full automatic warehouse VisualBasic 2010*, Beispielprogramm.

[3] Gail, Andreas: *Stichcode-Leser am Robo TX Controller (1): Automatisiert mit ROBO Pro*. ft:pedia 3/2014, S. 66-71.

[4] Gail, Andreas: *fischertechnik: Hochregallager mit camera barcode scanner und Lagerdatenbank*. Youtube-Video.

[5] Gail, Andreas: *RoboRISC: Visual Basic für den Robotics TXT.* In dieser ft:pedia.

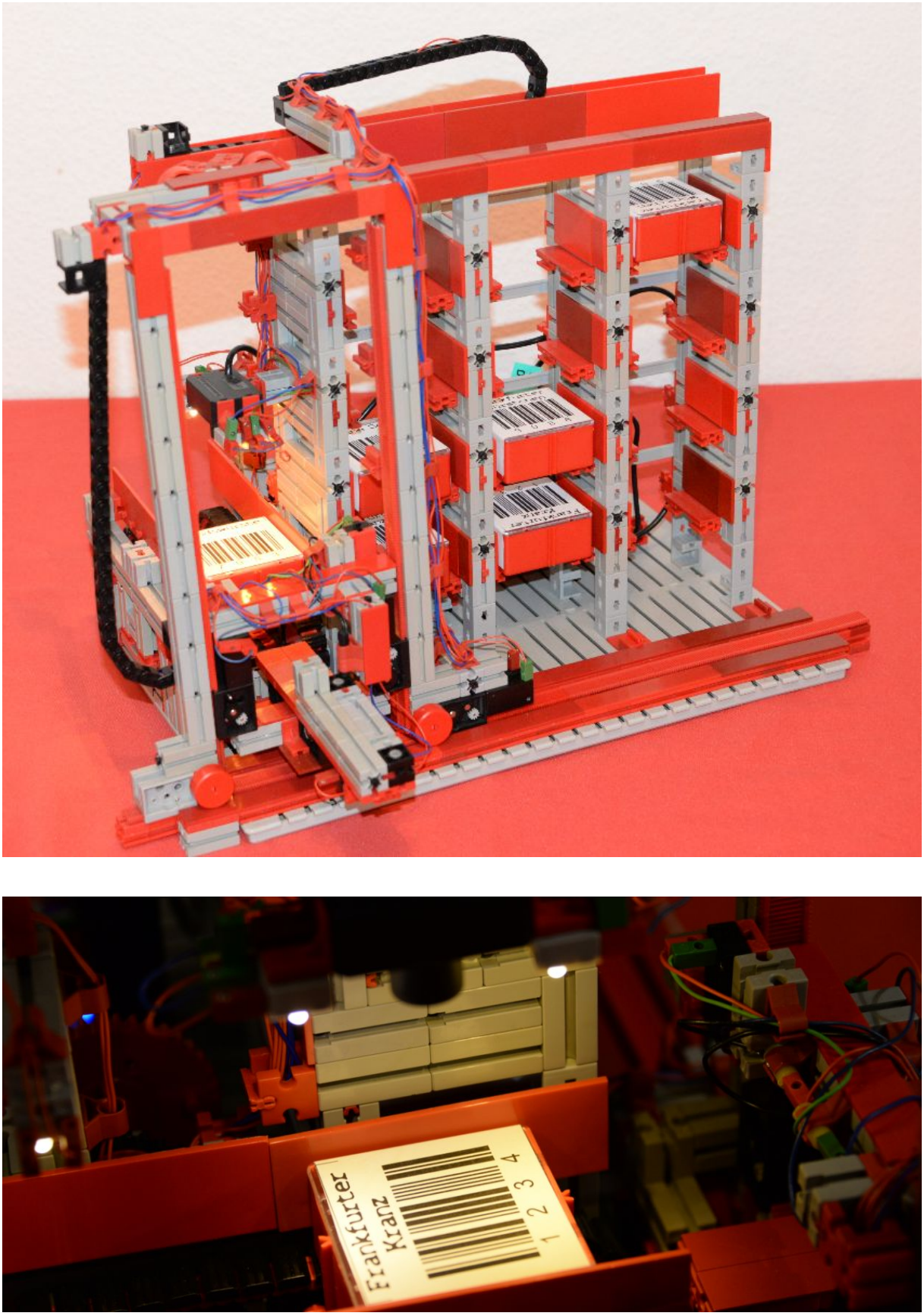


Abb. 10: Gesamtansicht (oben) und Strichcode Registrierung (unten)

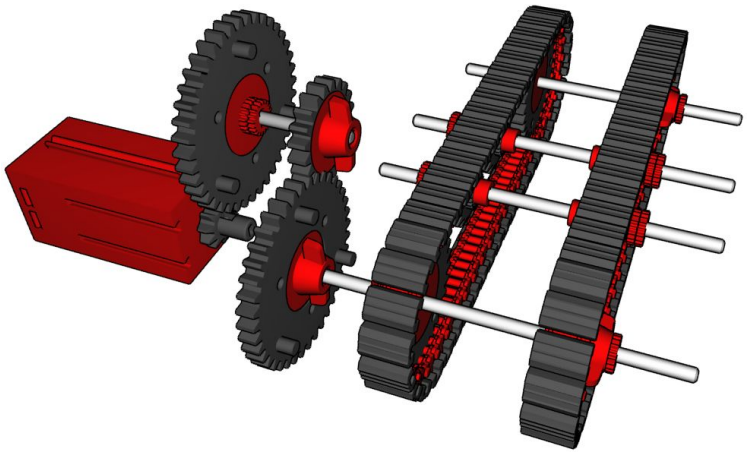


Abb. 11: Förderband mit Antrieb und untersetztem Getriebe

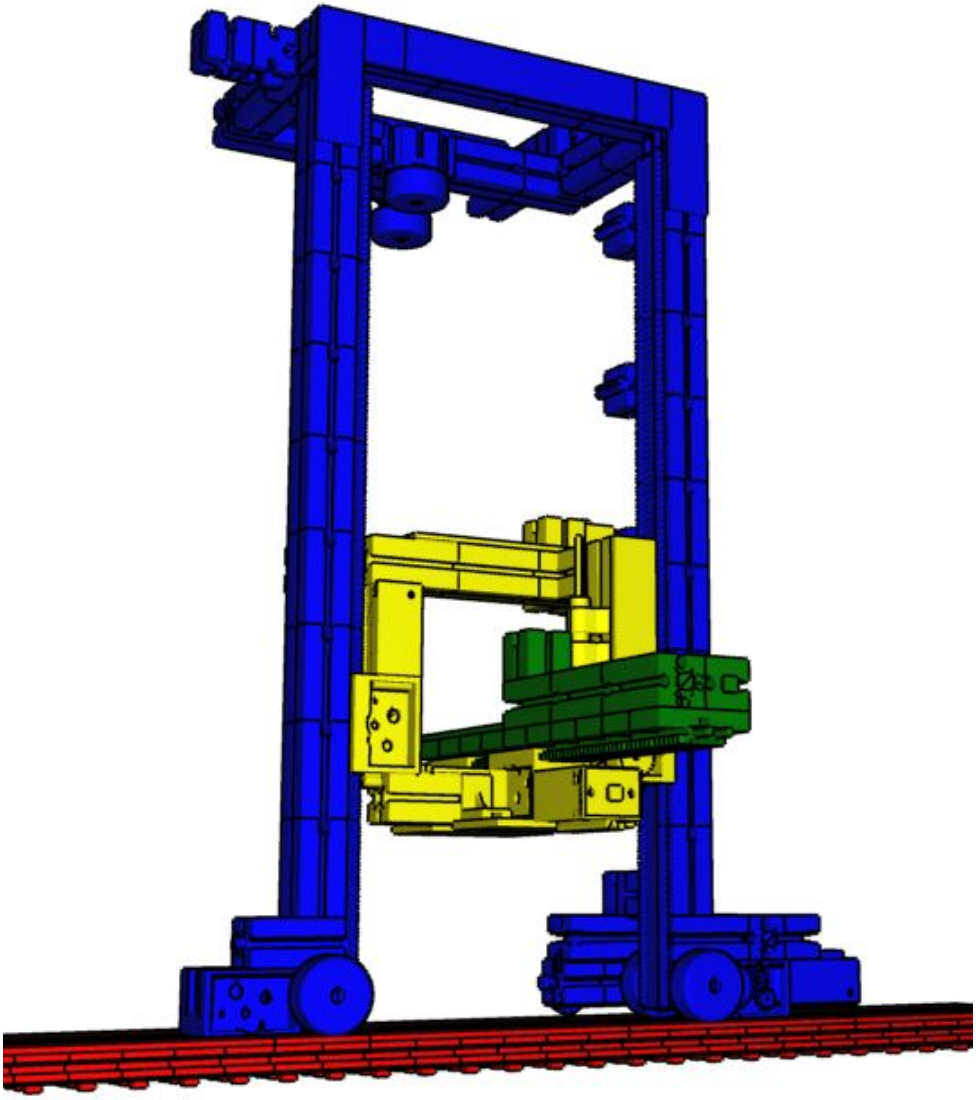


Abb. 12: Detaildarstellung des Laderoboters auf dem Fahrweg (rot), bewegte Teile in x-Richtung (blau), y-Richtung (gelb) und z-Richtung (grün)

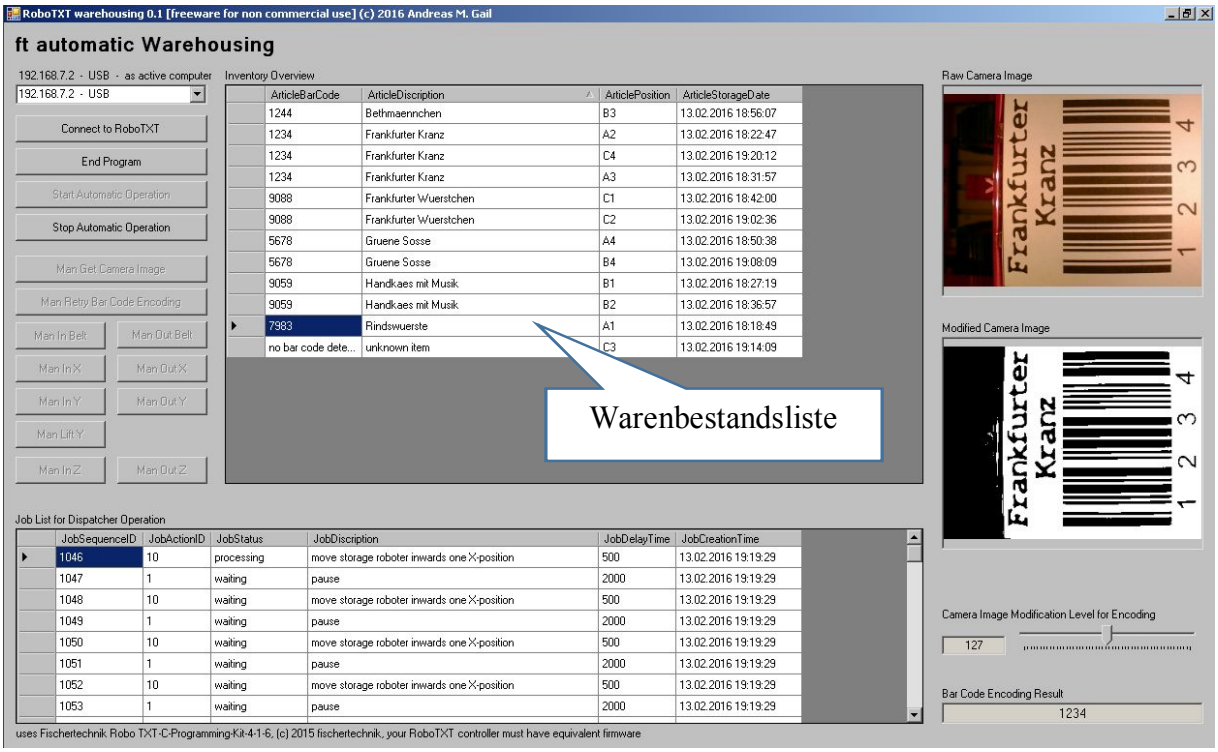


Abb. 13: PC-Bedienoberfläche

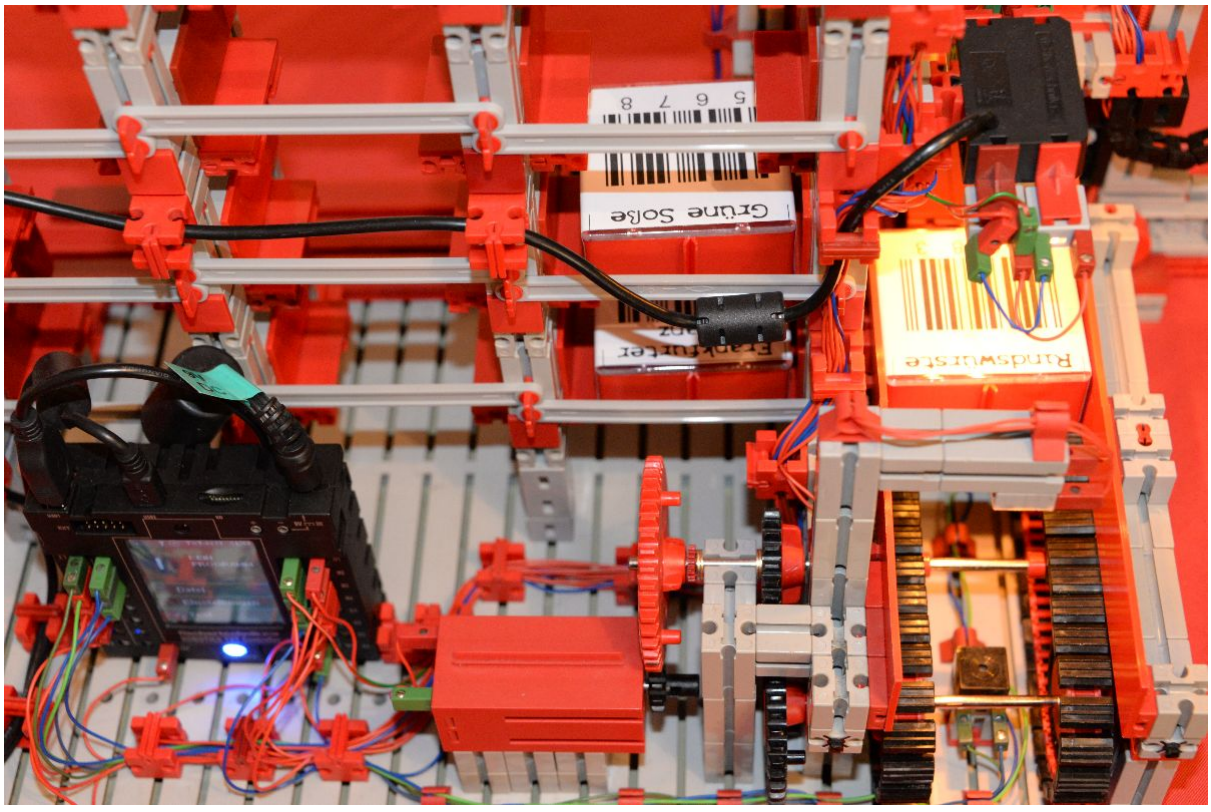


Abb. 14: Gesamtansicht von der Ladeseite

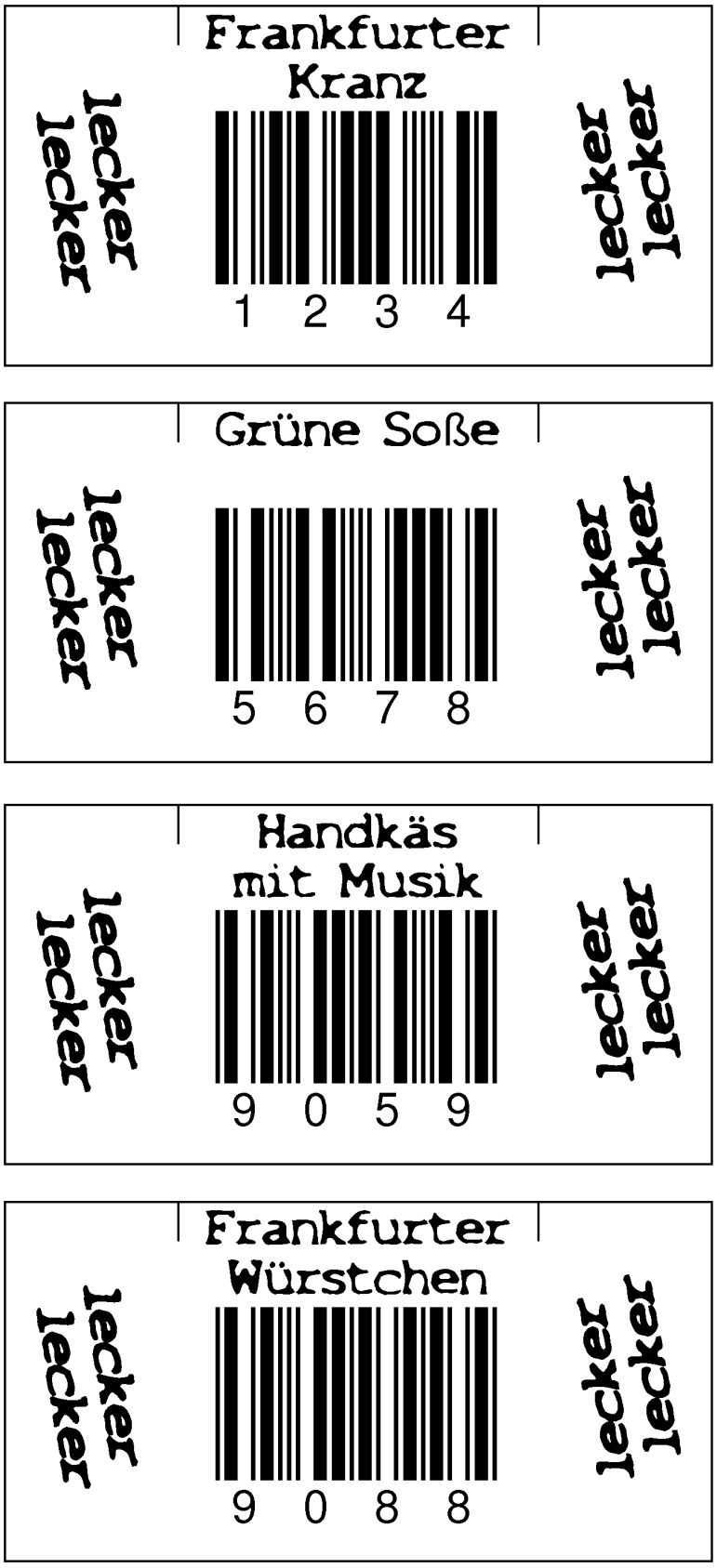


Abb.15: Beispiele Gebindeetiketten (Originalgröße 5,5cm x 10,8cm)